(12)

US005805846A

United States Patent [19]

Nakajima et al.

[11] Patent Number: 5,805,846

[45] Date of Patent: Sep. 8, 1998

[54] SYSTEM AND METHOD FOR DYNAMICALLY SHARING AN APPLICATION PROGRAM AMONG A PLURALITY OF CONFERENCE DEVICES WHILE MAINTAINING STATE

[75] Inventors: **Amane Nakajima**, Machida; **Makoto Kobayashi**; **Fumio Ando**, both of Kawasaki, all of Japan

[73] Assignee: **International Business Machines Corporation**, Armonk, N.Y.

[21] Appl. No.: **754,518**

[22] Filed: **Nov. 21, 1996**

Related U.S. Application Data

[63] Continuation of Ser. No. 371,915, Jan. 12, 1995, abandoned.

[30] **Foreign Application Priority Data**

Feb. 14, 1994 [JP] Japan 6-017269

[51] **Int. Cl.**^{6} **G06F 13/00**

[52] **U.S. Cl.** **395/330**; 395/200.04

[58] **Field of Search** 395/329–332, 395/200.04, 200.12, 330; 379/202

[56] **References Cited**

U.S. PATENT DOCUMENTS

5,195,086	3/1993	Baumgartner et al.	395/153
5,208,912	5/1993	Nakayama et al.	395/153
5,280,583	1/1994	Nakayama et al.	395/200
5,283,861	2/1994	Dangler et al.	395/153
5,353,398	10/1994	Kitahara et al.	395/153
5,363,507	11/1994	Nakayama et al.	395/153
5,379,374	1/1995	Ishizaki et al.	395/153
5,491,743	2/1996	Shiio et al.	379/202
5,491,798	2/1996	Bonsall et al.	395/200.04
5,533,183	7/1996	Henderson, Jr. et al.	395/344
5,537,548	7/1996	Fin et al.	395/200.04

OTHER PUBLICATIONS

Ensor et al., "Control issues in multimedia conferencing", Proceedings of TRICOMM '91: IEEE Conference on Communications Software: Communications for Distributed Applications and Systems, pp. 133–143, Apr. 1991.

Ahuja et al., "Coordination and control of multimedia conferencing", IEEE Communications Magazine, v. 30, n. 5, pp. 38–43, May 1992.

Ahuja et al., "A comparison of application sharing mechanisms in real–time desktop conferencing systems", Conf. on Office Information Systems, SIGOIS Bulletin, v. 11, n. 2–3, pp. 238–248, Apr. 1990.

Ahuja et al., "Networking requirements of the Rapport multimedia conferencing system", IEEE INFOCOM '88, pp. 746–751, Mar. 1988

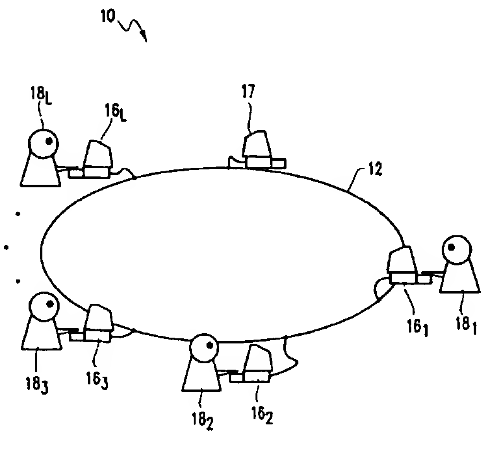
(List continued on next page.)

Primary Examiner—Joseph R. Burwell
Attorney, Agent, or Firm—Whitham, Curtis, Whitham & McGinn; Kevin M. Jordan, Esq

[57] **ABSTRACT**

A method for dynamically sharing an application in a conference system while maintaining state regardless of whether the application is under execution in all conference devices. The conference system includes a plurality of conference devices with at least one conference device executing an application program. The method for dynamically sharing the application program among the plurality of conference devices in the conference system while maintaining state, includes the steps of: communicating a request to share the application program from a requesting conference device which is executing the application to all enrolled conference devices; storing an executed state of the application program to be shared in the requesting conference device; starting the application program to be shared by enrolled conference devices in which an application program corresponding to the application program to be shared is not started; communicating the executed state stored in the requesting conference device to all enrolled conference devices; and processing by all enrolled conference devices of the application program to be shared in a shared executed state equal to the executed state communicated.

5 Claims, 14 Drawing Sheets



OTHER PUBLICATIONS

Ohmori et al., "Distributed cooperative control for sharing applications based on multiparty and multimedia desktop conferencing system: MERAID", Proceedings of the 12th Int'l Conf. on Distributed Computing Systems, pp. 538–546, Jun. 1992.

Maeno et al., "Distributed desktop conferencing system (MERMAID) based on group communication architecture", ICC '91, v. 1, pp. 520–525, Jun. 1991.

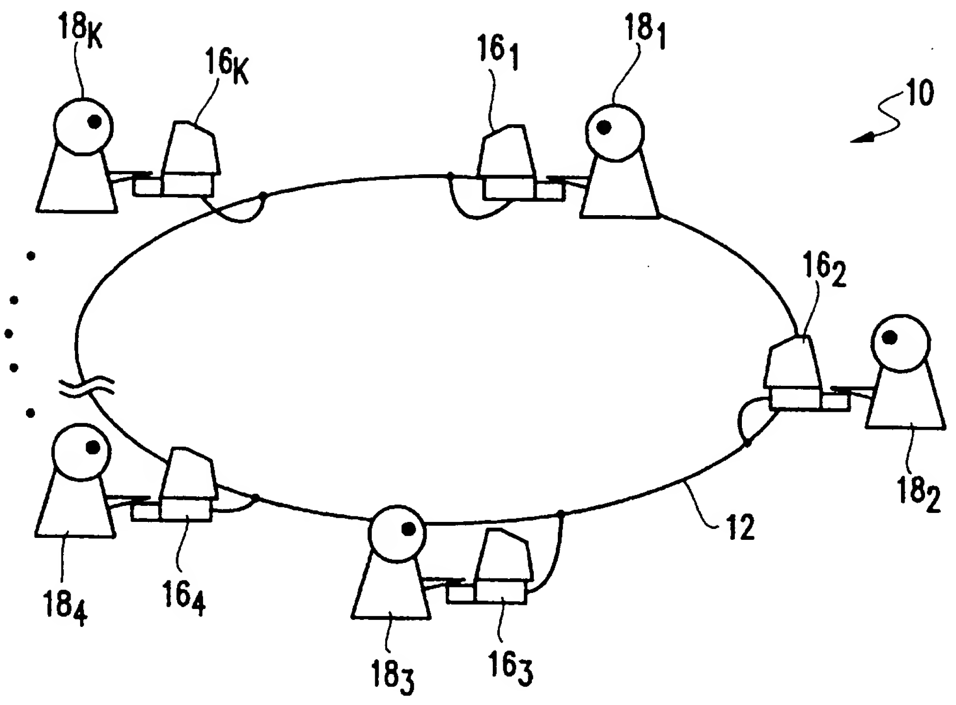
At&T Vistium™ Share Software User Guide, At&T Global Information Solutions, Ch.'s 4 & 5, pp. 4–1 to 4–20 & 5–1 to 5–26, Jul. 1994.

"Applications shared across HP. Sun, IBM and SGI", *HP Professional*, v. 8, n. 1, p. 58(1), Jan. 1994.

InSoft® Communique!™ User's Guide (version for UNIX); InSoft® Inc., Ch.'s 2,8, & 16, pp. 35–70, 131–162, and 261–268, Jan. 1994.

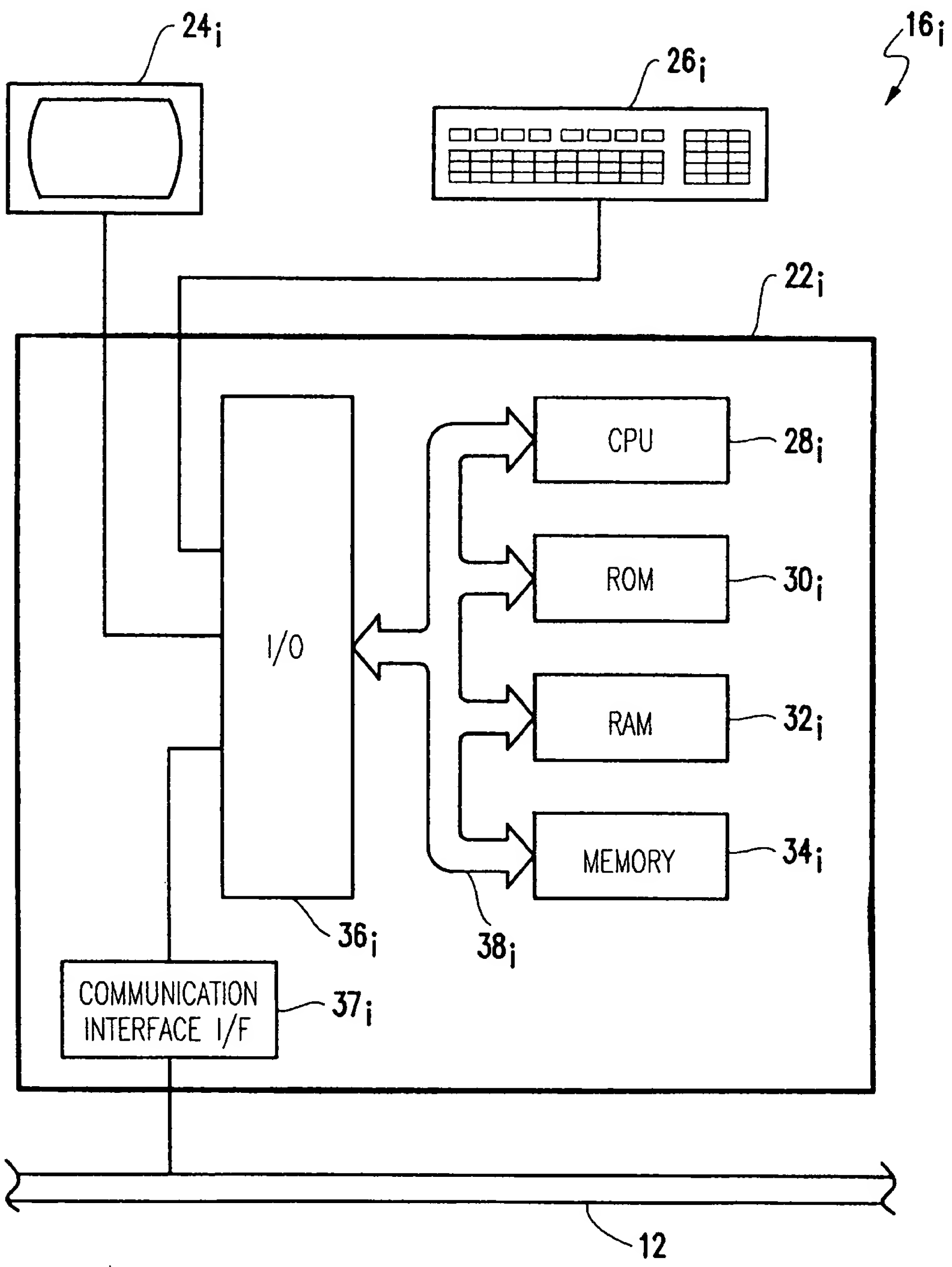
T. Ohmori et al., "Cooperative Control for Sharing Applications based on Distributed Multiparty Desktop Conferencing System: MERMAID", Supercomm—International Conference on Communications 92, vol. 2, 14 – 18 Jun. 1992, Chicago, pp. 1069–1075.

W. H. Leung et al., "Multimedia Conferencing Capabilities in an Experimental Fast Packet Network", Proceedings—International Switching Symposium 1992, vol. 2, 25 – 30 Oct. 1992, Yokohama, Japan, pp. 258–262.



10: CONFERENCE SYSTEM
12: NETWORK
16_1 TO 16_K : TERMINALS
18_1 TO 18_K : USERS

FIG.1



24_i
26_i
16_i
22_i
CPU
28_i
ROM
30_i
I/O
RAM
32_i
MEMORY
34_i
36_i
38_i
COMMUNICATION INTERFACE I/F
37_i
12

FIG.2

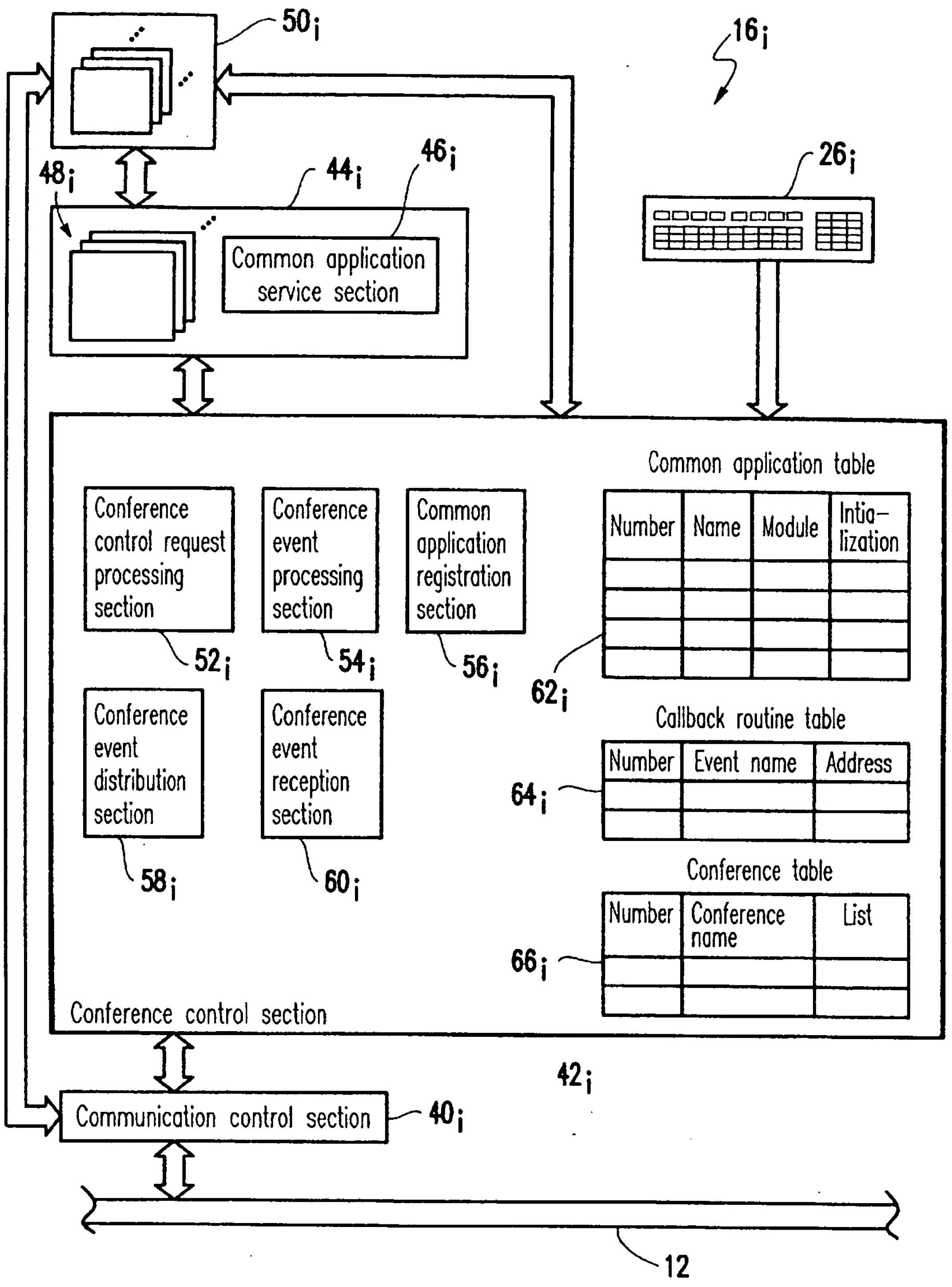

50$_i$
16$_i$
26$_i$
48$_i$
44$_i$
46$_i$
Common application service section
Common application table
Number
Name
Module
Intia-lization
Conference control request processing section
Conference event processing section
Common application registration section
52$_i$
54$_i$
56$_i$
62$_i$
Callback routine table
Number
Event name
Address
Conference event distribution section
Conference event reception section
64$_i$
58$_i$
60$_i$
Conference table
Number
Conference name
List
66$_i$
Conference control section
42$_i$
Communication control section
40$_i$
12

FIG.3

Conference event CE

(Conference Events)

Conference event		[Event name]	<Parameter>
Conference processing termination		TERMINATE	(none)
Basic	New	NEW	toolNumber, toolID
	Opening	OPEN	toolNumber, toolID
	Closing	CLOSE	toolNumber, toolID
	Deletion	DELETE	toolNumber, toolID, confID
Sharing	Start of assignment	BEGINSHARE	toolNumber, toolID, confID
	End of sharing	ENDSHARE	toolNumber, toolID, confID
Partici-pant	Addition	ENROLLED	confID, userID
	Deletion	UNENROLLED	confID, userID
New participation and exiting	Start of participation	ENTER-START	confID, userID
	Completion	ENTER-COMPLETE	confID, userID
	Start of Existing	LEAVE-START	confID, userID
	Completion	LEAVE-COMPLETE	confID, userID
Control Unit	Registration	FLOOR-REGISTERED	toolNumber, confID, floorID
	Erasure	FLOOR-UNREGISTERED	toolNumber, confID, floorID
	Arrangement	FLOOR-ENQUEUED	toolNumber, confID, floorID, userID
	Release	FLOOR-RELEASED	toolNumber, confID, floorID, userID
	Acquisition	FLOOR-OBTAINED	toolNumber, confID, floorID, userID
	Cancel	FLOOR-CANCELLED	toolNumber, confID, floorID, userID

FIG.4

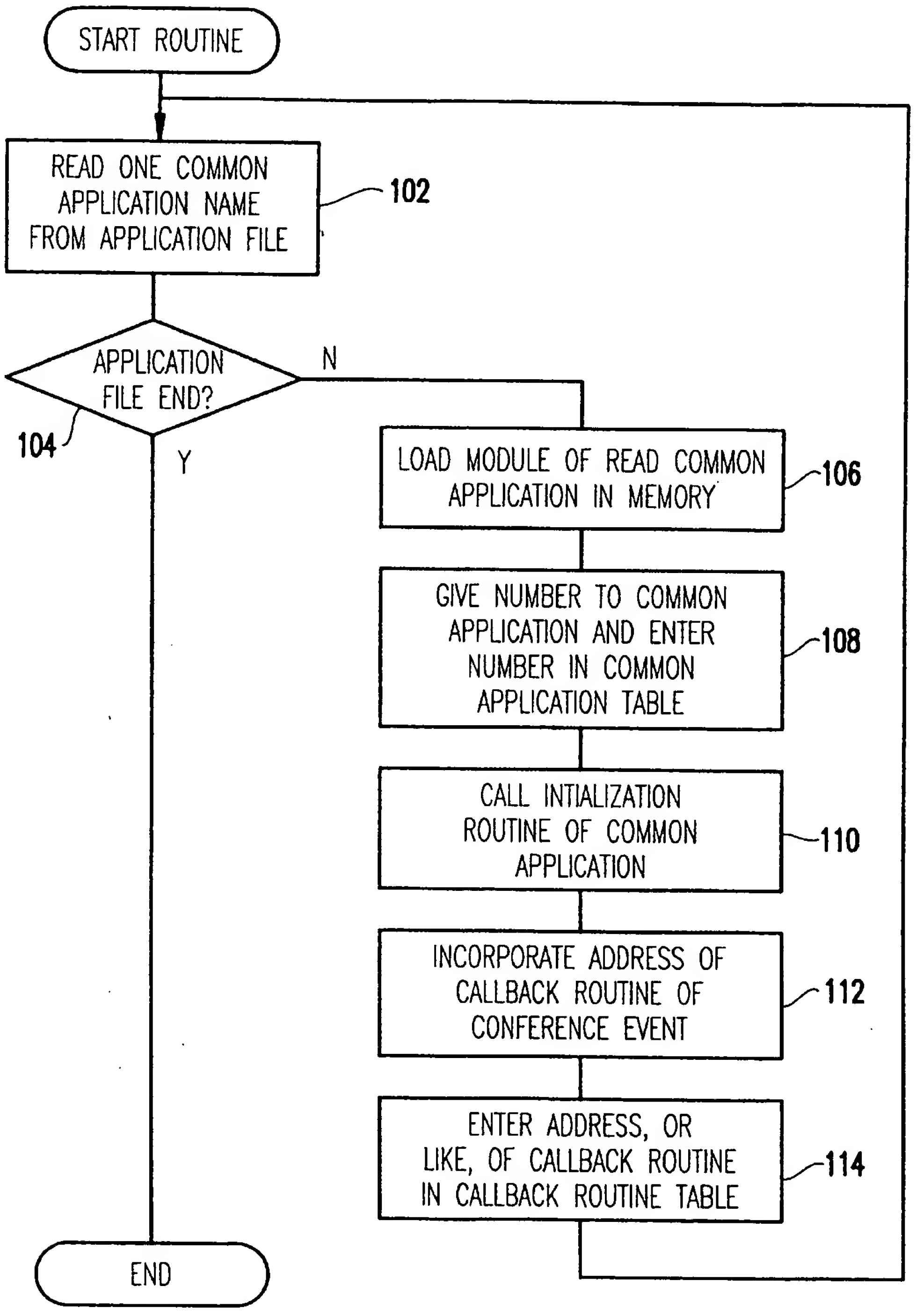

START ROUTINE
READ ONE COMMON APPLICATION NAME FROM APPLICATION FILE
102
APPLICATION FILE END?
N
104
Y
LOAD MODULE OF READ COMMON APPLICATION IN MEMORY
106
GIVE NUMBER TO COMMON APPLICATION AND ENTER NUMBER IN COMMON APPLICATION TABLE
108
CALL INTIALIZATION ROUTINE OF COMMON APPLICATION
110
INCORPORATE ADDRESS OF CALLBACK ROUTINE OF CONFERENCE EVENT
112
ENTER ADDRESS, OR LIKE, OF CALLBACK ROUTINE IN CALLBACK ROUTINE TABLE
114
END

FIG.5

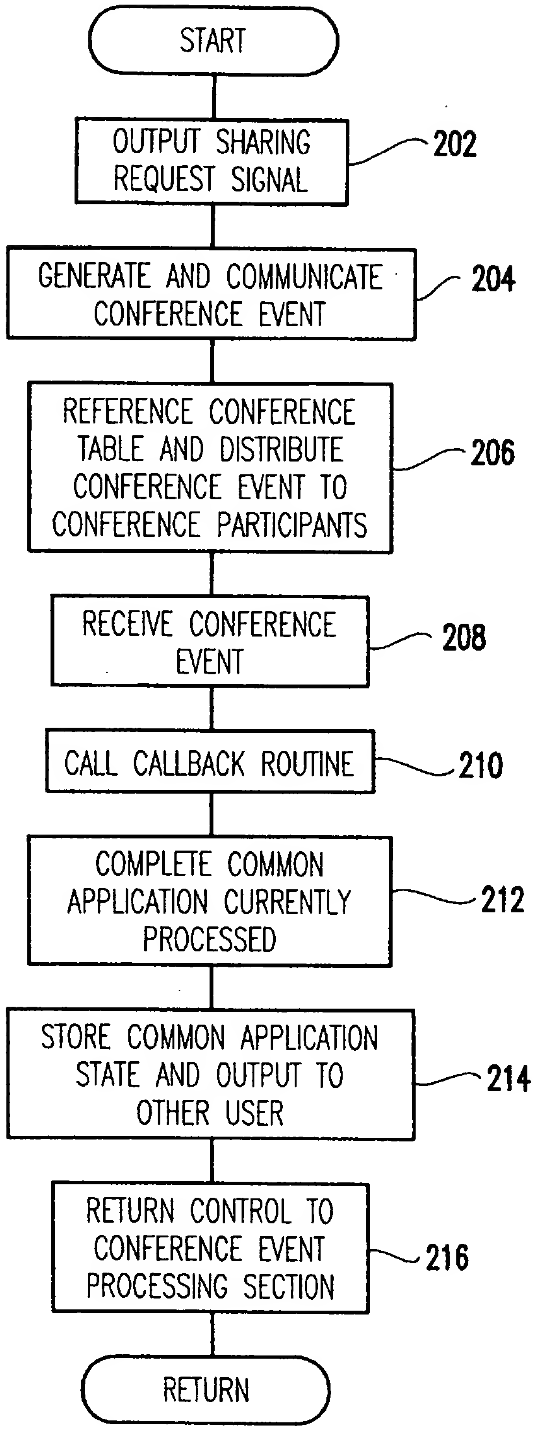

START
OUTPUT SHARING REQUEST SIGNAL
202
GENERATE AND COMMUNICATE CONFERENCE EVENT
204
REFERENCE CONFERENCE TABLE AND DISTRIBUTE CONFERENCE EVENT TO CONFERENCE PARTICIPANTS
206
RECEIVE CONFERENCE EVENT
208
CALL CALLBACK ROUTINE
210
COMPLETE COMMON APPLICATION CURRENTLY PROCESSED
212
STORE COMMON APPLICATION STATE AND OUTPUT TO OTHER USER
214
RETURN CONTROL TO CONFERENCE EVENT PROCESSING SECTION
216
RETURN

FIG.6

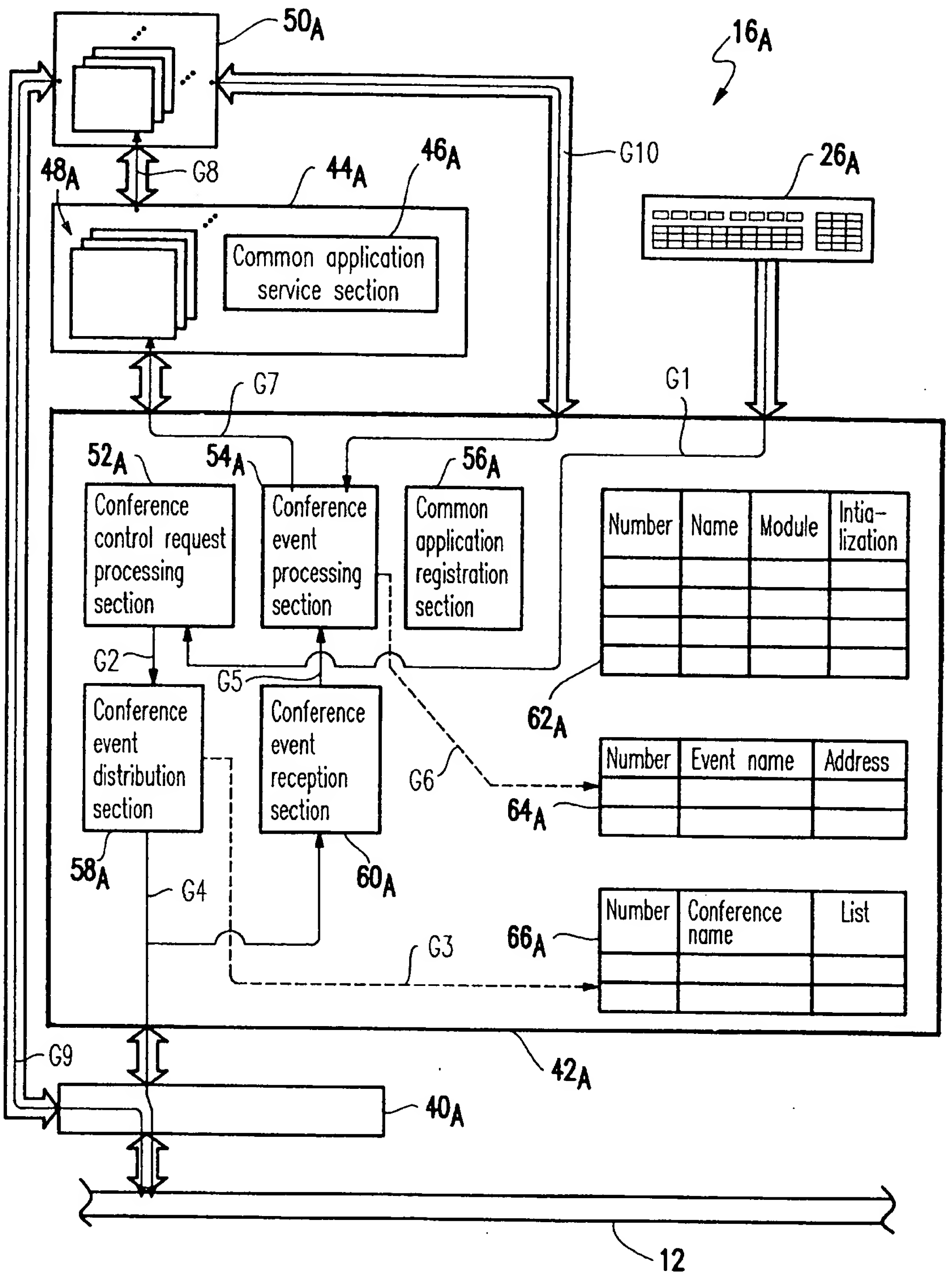

50A
16A
48A
G8
44A
46A
Common application service section
G10
26A
G7
G1
52A
54A
56A
Conference control request processing section
Conference event processing section
Common application registration section
Number
Name
Module
Intia-lization
G2
G5
Conference event distribution section
Conference event reception section
62A
G6
Number
Event name
Address
64A
58A
G4
60A
Number
Conference name
List
66A
G3
G9
42A
40A
12

FIG.7

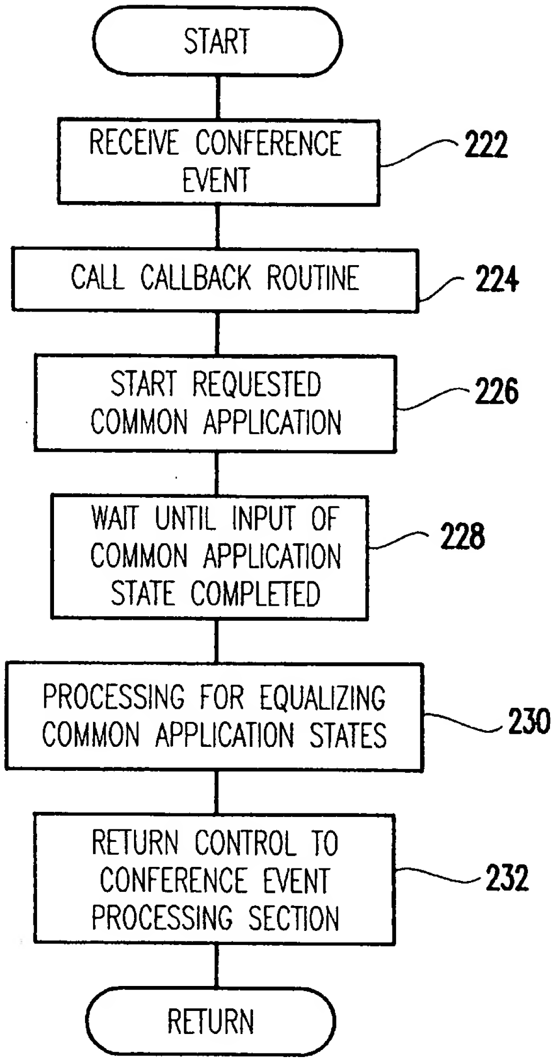
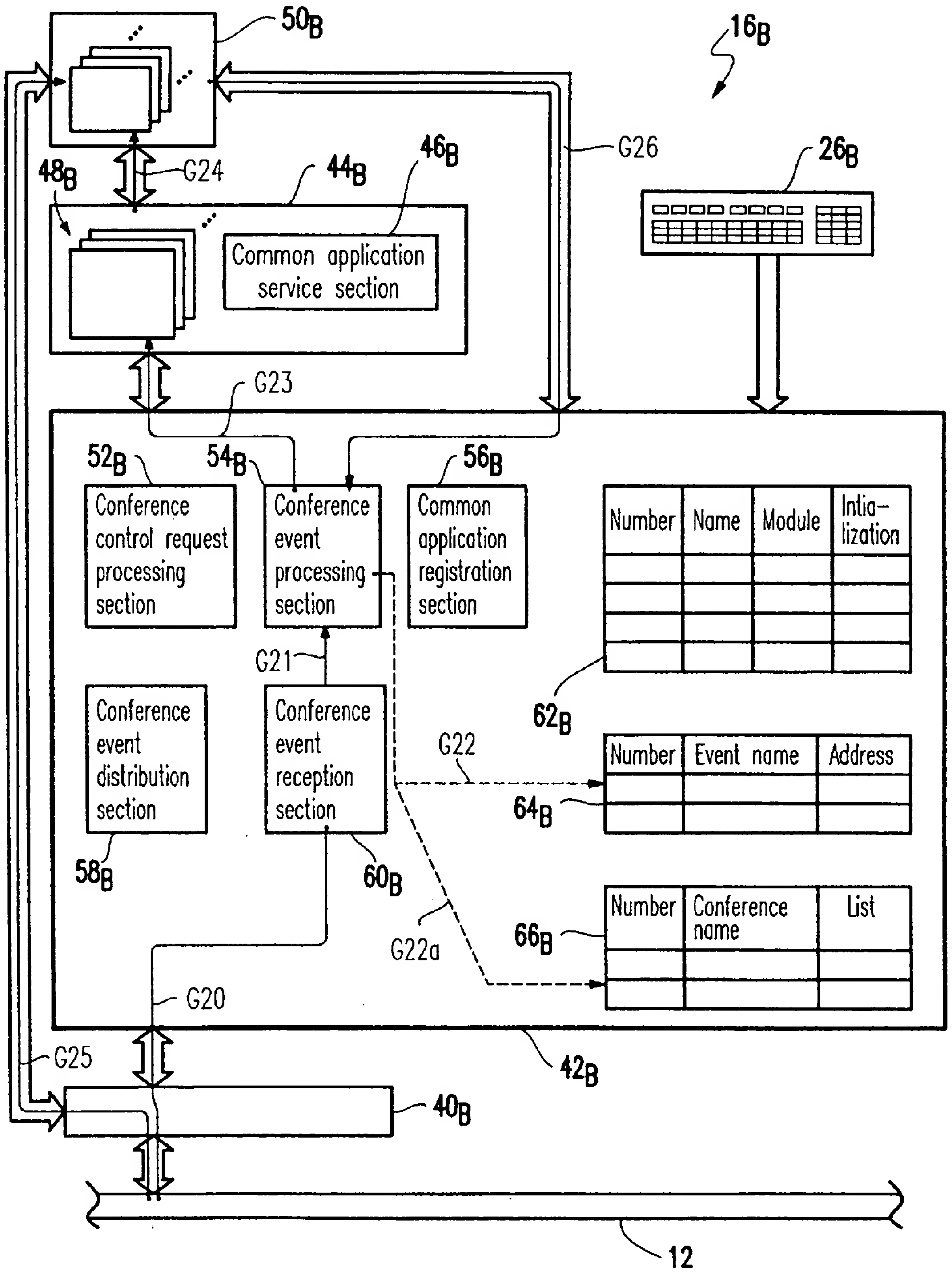

START
RECEIVE CONFERENCE EVENT
222
CALL CALLBACK ROUTINE
224
START REQUESTED COMMON APPLICATION
226
WAIT UNTIL INPUT OF COMMON APPLICATION STATE COMPLETED
228
PROCESSING FOR EQUALIZING COMMON APPLICATION STATES
230
RETURN CONTROL TO CONFERENCE EVENT PROCESSING SECTION
232
RETURN

FIG.8



50_B
16_B
G26
26_B
48_B
G24
44_B
46_B
Common application service section
G23
52_B
54_B
56_B
Conference control request processing section
Conference event processing section
Common application registration section
Number
Name
Module
Intia-lization
G21
62_B
Conference event distribution section
Conference event reception section
G22
Number
Event name
Address
64_B
58_B
60_B
Number
Conference name
List
66_B
G22a
G20
G25
42_B
40_B
12

FIG.9

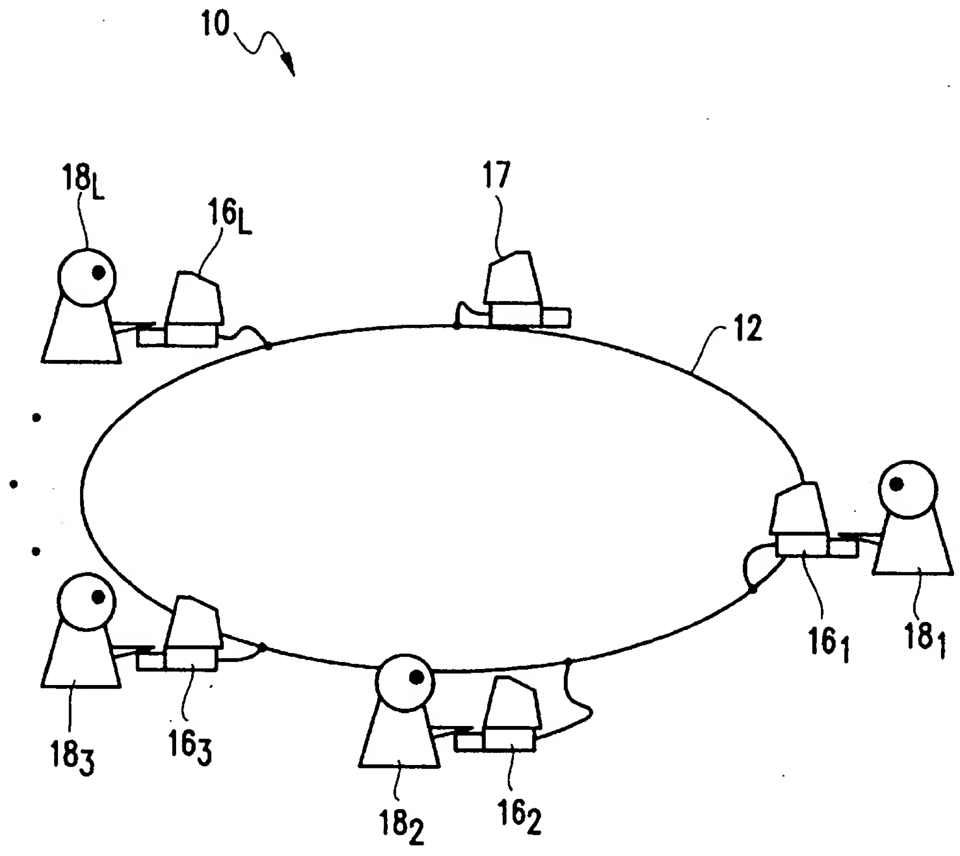
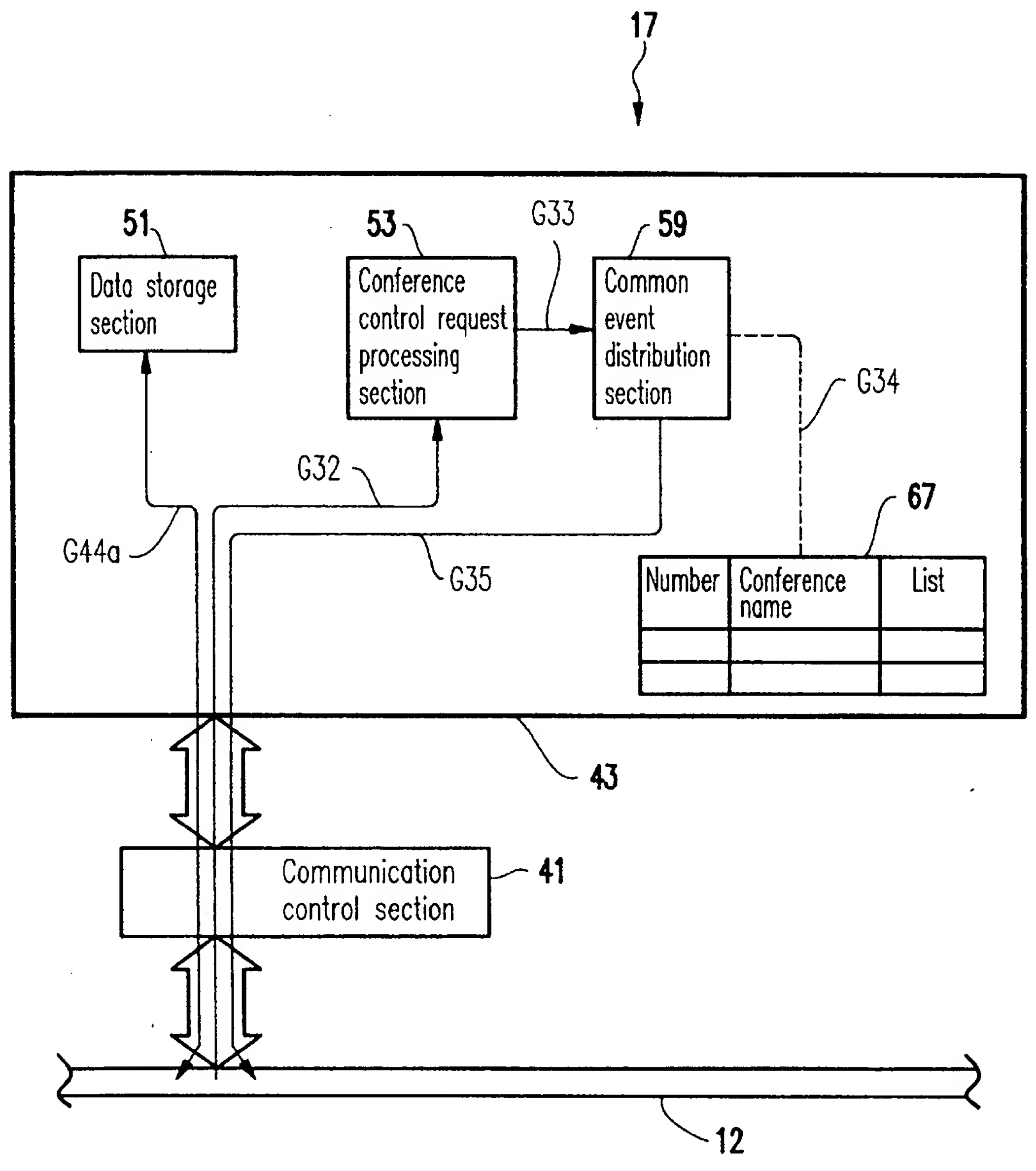


FIG.10



17
51
Data storage section
53
Conference control request processing section
G33
59
Common event distribution section
G34
G32
67
G44a
G35
Number
Conference name
List
43
Communication control section
41
12

FIG.11

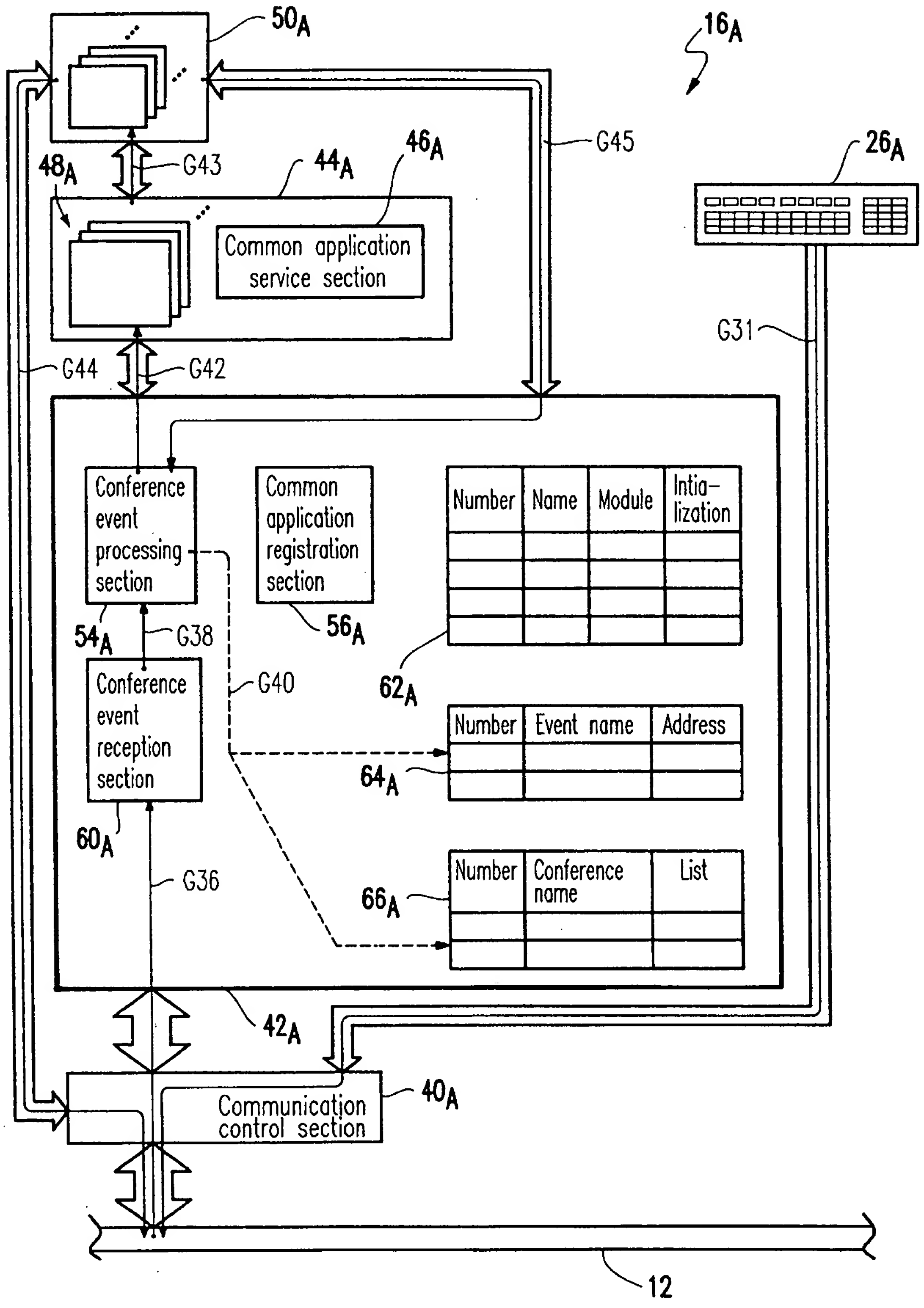

50A
16A
G45
46A
48A
G43
44A
26A
Common application service section
G31
G44
G42
Conference event processing section
Common application registration section
Number
Name
Module
Intia-lization
54A
G38
56A
62A
G40
Conference event reception section
Number
Event name
Address
64A
60A
G36
66A
Number
Conference name
List
42A
40A
Communication control section
12

FIG. 12

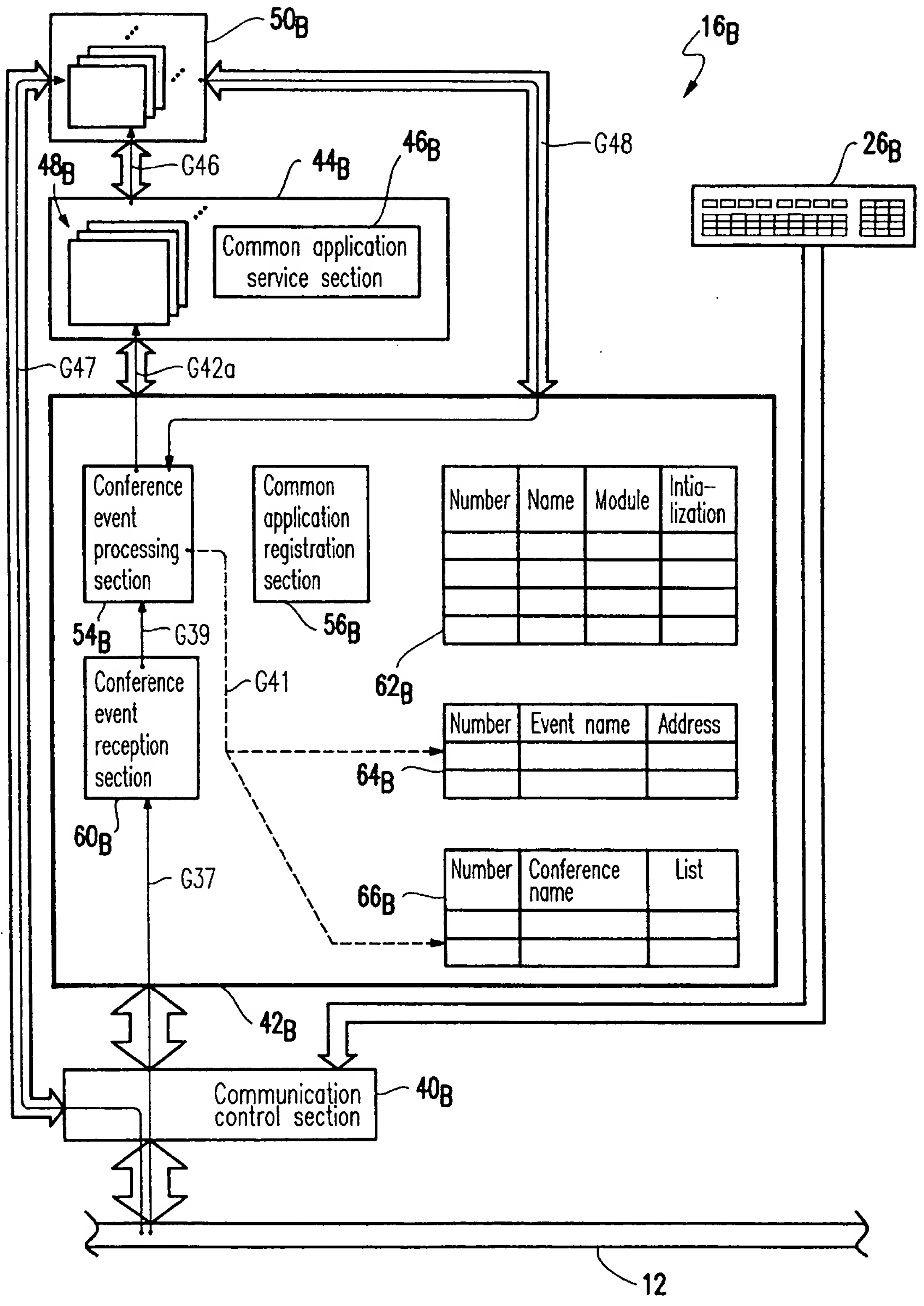
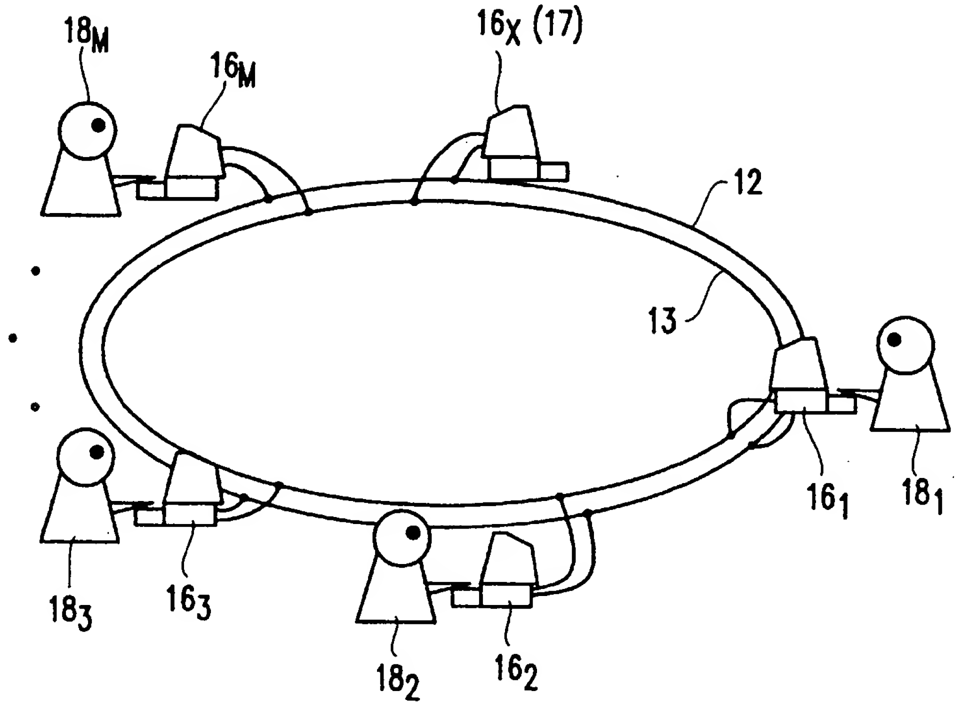
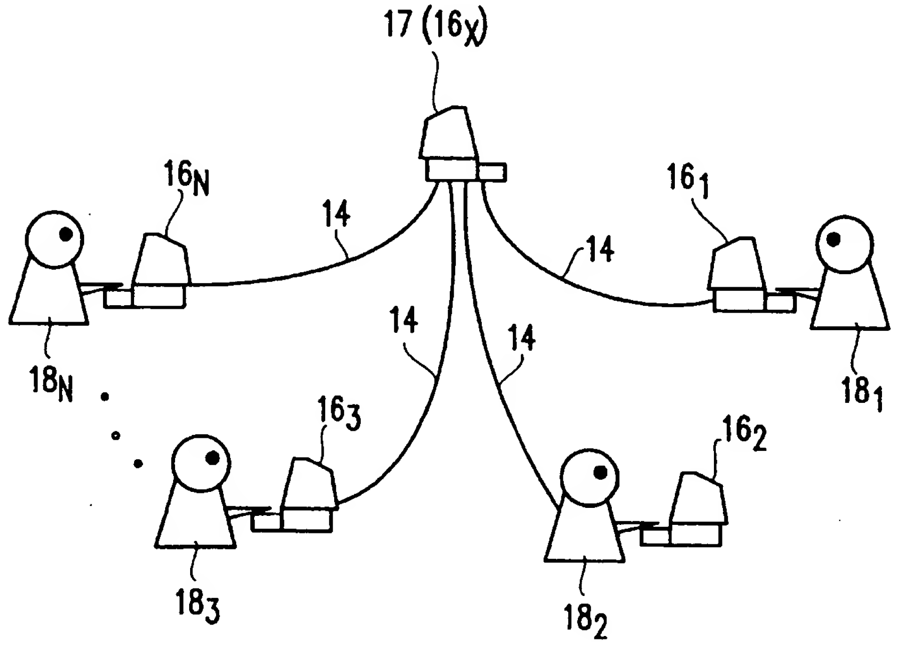

50_B
16_B
G48
G46
44_B
46_B
48_B
26_B
Common application service section
G47
G42a
Conference event processing section
Common application registration section
Number
Name
Module
Intia-lization
54_B
G39
56_B
G41
62_B
Conference event reception section
Number
Event name
Address
64_B
60_B
G37
66_B
Number
Conference name
List
42_B
40_B
Communication control section
12

FIG.13



18_M
16_M
16_X (17)
12
13
16_1
18_1
18_3
16_3
18_2
16_2

FIG.14



17 (16_X)
16_N
14
14
16_1
14
14
18_N
16_3
16_2
18_1
18_3
18_2

FIG.15

SYSTEM AND METHOD FOR DYNAMICALLY SHARING AN APPLICATION PROGRAM AMONG A PLURALITY OF CONFERENCE DEVICES WHILE MAINTAINING STATE

RELATED APPLICATIONS

This application is a continuation of U.S. patent application Ser. No. 08/371,915, filed Jan. 12, 1995, now abandoned.

FIELD OF THE INVENTION

The present invention relates to a conference system control method, a conference device, and a conference system, and, more particularly, to a conference system control method, a conference device, and a conference system for opening a conference by using a plurality of conference devices connected by a network or communication line such as a LAN, ISDN, or telephone line.

BACKGROUND OF THE INVENTION

In a conference system, each of a plurality of users connects each terminal to a line or network to open a conference while transferring data between terminals. In this type of system, conference systems for realizing an application program (hereafter referred to as an application) which can be shared by all users are roughly divided into "centralized" and "decentralized."

The centralized system allows data to be input to an application from a plurality of user terminals and transmits output data to all user terminal. Rapport uses a centralized system (S. R. Ahuja, J. R. Ensor, and D. N. Horn, "The Rapport Multimedia Conferencing System," Proc. COIS 88, pp. 1–8, 1988).

However, the centralized system is not preferable because much data is output from the application, increasing the load on hardware. Moreover, the centralized system is restricted in that the system must be applied to a server-client window system in which display processing and control are independently performed like the X-WINDOW SYSTEM which operates on UNIX.

The decentralized system ensures the same state by executing the same application in each user terminal and cooperatively operating each application. The system is used for ASSOCIA (Yoshiyuki Nakayama et al., "A Computer-Supported Multiparticipant Realtime Teledialogue System," Thesis, Journal of the Information Processing Society of Japan, Vol. 32, No. 9, pp. 1190–1199, 1991) and MMConf (T. Crowley et al., "MMConf: An Infrastructure for Building Shared Multimedia Applications," Proc. CSCW 90, pp. 329–342, 1990).

ASSOCIA keeps a plurality of instances (entities) of an application in the same state by ensuring at the system side that all data values input to all applications from a plurality of users coincide with each other in a time series. Therefore, it is possible to share the "instances" of all applications and keep them in the same state when start and execution are simultaneously performed in a plurality of user terminal.

With ASSOCIA, however, when a user by whom an application currently executed is not shared is going to share the application, the state of an application of a user who already shares the application may not coincide with the state of an application of a user who is going to share the application in the middle. That is, the application currently executed has already-read execution files and states such as

attribute values of the present application (e.g. cursor location, expansion or contraction, and fonts). Therefore, even if the same applications are started by other user terminals at the initial state and the same input is given to applications after they are shared, it is impossible to realize a shared application which coincides with any other application because each state when the application is shared differs. Therefore, to keep a currently-executed application in a consistent state when the application is shared, a low-level consistency of giving the same input is insufficient.

Also, in the case of MMConf, it is possible to keep a plurality of manifestations of applications simultaneously started in the same state but it is impossible to keep a shared application consistent because processing for sharing an application is not assumed.

As another example of the above-mentioned, the official gazette of Japanese Patent Laid-Open No. Hei 3-192845 discloses a method for controlling an interaction system serving as a conference system, which realizes a dialogue by the fact that a plurality of users accesses a network of terminals. This method realizes a dialogue equivalent to a conference by processing participation in the dialogue and the exit of a user who can attend a conference (enrollee) from the dialogue by each terminal so that participation and exiting can be done easily.

However, the above interaction system control method only processes the participation in and exit from a dialogue equivalent to a conference. Therefore, it is impossible to keep a shared application consistent at the start of a conference and during the conference because processing for sharing an application is not assumed.

Moreover, the official gazette of Japanese Patent Laid-Open No. Hei 4-186456 discloses a common information processing system serving as a conference system making it possible for a plurality of users to perform a dialog and common information processing by connecting terminal to a network or line. This system allows each user to process electronic information in common with other terminals by performing a dialogue while referencing the same output result and so on.

However, the common information processing system also has a restriction in that the system must be applied to a server-client window system in which display processing and control are independently performed like the X-WINDOW SYSTEM which operates on UNIX.

Thus, there is a need for a more cost-effective conference system and method which allows dynamic sharing of applications regardless of whether a user is currently executing the application. The present invention addresses such a need.

SUMMARY OF THE INVENTION

It is therefore an object of the present invention to provide a conference system whereby all user terminals executing a common application are maintained in a same state.

It is yet another object of the present invention to bring user terminals joining a conference already in progress to a same state as other user terminals in the conference.

According to the invention, a plurality of individual user terminals are joined together by a network. In operation, a requesting user terminal runing an application program requests a conference. While the application program may be of any type, for the purposes of illustration, a blackboard program is run whereby conferees attending the conference can write on the blackboard and erase the blackboard just as a conferee might do when physically attending a conference in a conventional conference room. The requesting user terminal issues a request to share the application program with all conferee user terminals causing all conferee user terminals to load the application program. The requesting user terminal then transmits the current state of the application program to the conferee user terminals such that all user terminals are brought to the same state and the conference can proceed with all conferees seeing the same things on their individual terminals.

A callback protocol memory is also provided for storing a processing program and identifiers corresponding to event information whereby when a state of the application is changed, such as by, for example, a conferee writing a message on the blackboard, the new state of the application is transmitted to all user terminals participating in the conference such that all such terminals maintain a same state.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a schematic block diagram showing the conference system of the first embodiment;

FIG. 2 is a schematic block diagram showing a terminal of the first embodiment;

FIG. 3 is a block diagram showing the rough internal constitution of the computer of a terminal of the first embodiment;

FIG. 4 is a corresponding diagram showing types of conference events, names, and parameters;

FIG. 5 is a flowchart showing the flow of a start for dynamically incorporating a common application;

FIG. 6 is a flowchart showing the flow of processing in a requesting terminal when a plurality of users shares a common application;

FIG. 7 is a conceptual view showing the flow of processing in a requesting terminal when a plurality of users shares a common application;

FIG. 8 is a flowchart showing the flow of processing in a requested terminal when a plurality of users shares a common application;

FIG. 9 is a conceptual view showing the flow of processing in a requested terminal when a plurality of users shares a common application;

FIG. 10 is a schematic block diagram showing the conference system of the second embodiment;

FIG. 11 is a conceptual view showing the flow of processing in an integrated controller when a plurality of users shares a common application in the second embodiment;

FIG. 12 is a conceptual view showing the flow of processing in a request-side terminal unit when a plurality of users shares a common application in the second embodiment;

FIG. 13 is a conceptual view showing the flow of processing in a requested terminal when a plurality of users shares a common application in the second embodiment;

FIG. 14 is a schematic block diagram showing a conference system for separately sending or receiving voice and animation information; and

FIG. 15 is a schematic block diagram showing a different connection configuration among a plurality of terminals in a conference system.

Description of Symbols

10	Conference system
12	Network
16_1 to **16_K**	Terminal units
18_1 to **18_K**	Users

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

Embodiments of the present invention are described below by referencing the accompanying drawings.

As shown in FIG. 1, a conference system **10** which is the first embodiment comprises an annular network **12** for sending/receiving a digital signal and K (a natural number) terminals **16_1** to **16_K** connected to the network **12**. In the conference system **10**, transfer data is transmitted through the network **12**. Each terminal **16_1** to **16_K** is operated by users **18_1** to **18_K**. The network **12** is not restricted to the above networks (Ethernet and Token Ring Network). It is possible to use a public telephone network, a network such as ISDN, or a combination of them as the network **12**. Moreover, each terminal **16_1** to **16_K** stores a conference room system **11** as an execution program for progressing a conference through terminals among a plurality of users as described later.

As shown in FIG. 2, the terminal **16_i** ($i{:}1 \leq i \leq K$) comprises a personal computer (hereafter referred to as a PC) **22_i** serving as a controller, a display **24_i** serving as a display, and keyboard **26_i** serving as an input unit. In addition to the above constitution, it is possible to connect a liquid crystal display or projector serving as a display, a mouse serving as a coordinate input unit, a camera serving as an image input unit, a microphone serving as a voice input unit, and a pair of speakers serving as voice output units. The microphone and the two speakers can be replaced with a head set such as a telephone set or transceiver. Moreover, it is possible to set the two speakers so as to produce a stereophonic effect or it is possible to use three or more speakers for multichannel reproduction or use one speaker for monaural reproduction. Input processing can be made on a CRT without using a keyboard by connecting a mouse and voices can be input or output by connecting a microphone and a speaker. In addition to the above constitution, it is possible to connect a digitizer serving as a coordinate input unit, a printer serving as an output unit, and a scanner serving as an input unit.

A PC **22_i** comprises a CPU **28_i**, ROM **30_i**, RAM **32_i**, memory **34_i**, and an input/output port I/O **36_i** connected by a bus **38_i** so that data and commands can be transferred between them. Memory **34_i** can use external memory such as a floppy disk drive, hard disk drive, or optoelectronic memory. The input/output port I/O **36_i** is connected to the network **12** through a communication interface I/F **37_i** and also connected to the display **24_i** and keyboard **26_i**.

In FIG. 3, the outline of the internal constitution of the PC **22_i** for realizing the conference room system **11** in the terminal **16_i** is shown in the form of a block diagram for each function. The PC **22_i** has a communication control section **40_i** for communicating data or the like with another PC$_j$ ($j{:}1 \leq j \leq K$, $i \neq j$), a conference control section **42_i** (described later) for controlling conference operations such as opening and closing of a conference (hereafter referred to as a conference room) opened among a plurality of terminal and an enrollee's entering and exit the room, a common application section **50_i** comprising one or more programs which are shared by enrollees in the conference room and concurrently used (e.g., a program or the like of a common blackboard on which a character or a diagram can be written or on which a character or diagram can be erased), and a common application management section **44_i**. The common application management section 44_i includes a callback routine section 48_i comprising a processing program (hereafter referred to as a callback routine) to the common application section 50_i for various instance processing (hereafter referred to as conference events; described later) in the case of the call of a common application service section 46_i, and the common application section 50_i or operations of a conference. The common application service section 46_i stores a processing routine included in a common application (e.g., a program for storing a state of a common application), which is called by an individual common application or by the common application service section 46_j ($i \neq j$) of another terminal.

The communication control section 40_i is connected to the network **12** in order to exchange data with a plurality of terminal used for one conference through a wide-area network such as a LAN or ISDN and also connected to the conference control section 42_i and the common application section 50_i. The communication control section may perform one-to-one data transfer to and from another terminal separately from an opened conference. The conference control section 42_i is connected to the common application section 50_i, common application management section 44_i, and keyboard 26_i. The common application section 50_i is connected to the common application management section 44_i.

The conference control section 42_i is a control section for processing a request for generation or deletion of a conference room, communicating with the conference control section 42_j of another terminal to process the acceptance of the request, and performing processing so as to synchronously keep the same contents relating to new enrollees in a conference and to the fact that a user having an application operating sends a conference event such as replacement or the like to the common application section 50_i. For example, the section **42***j* manages a conference member as a enrollee to a conference and the common-application use right and has a function for notifying the common application of a change in the number of conference enrollees, of moving the use right to a conference enrollee as a conference event.

Thus, for the conference system **11** of this embodiment, the conference control section 42_i for operating a conference and the common application section 50_i which is a conference event section simultaneously used by enrollees in a conference and stores common applications are independently constituted.

As shown in FIG. **4**, the conference event CE shows details of conference state change. The conference event CE comprises an event name and a parameter. Corresponding to the conference event CE, a callback routine for processing the conference event CE concerned is previously registered in the callback routine section 48_i. In FIG. **4**, event names TERMINATE, NEW, OPEN, CLOSE, and DELETE are shown as the conference event CE for basic conference operations such as termination of conference processing, new conference, opening of conference, closing of conference, and deletion of conference; event names BEGINSHARE and ENDSHARE are shown as the conference event CE for sharing a requested common application such as beginning of assignment and ending of assignment for sharing; event names ENROLLED and UNENROLLED are shown as the conference event CE for changing enrollees in a conference room such as addition and deletion of enrollees; event names ENTER-START, ENTER-COMPLETE, LEAVE-START, and LEAVE-COMPLETE are shown as the conference event CE for newly participating and exiting a conference room such as new participation

and exiting; and event names FLOOR-REGISTERED, FLOOR-UNREGISTERED, FLOOR-ENQUEUED, FLOOR-RELEASED, FLOOR-OBTAINED, AND FLOOR-CANCELED are shown as the conference event relating to the conference room control right such as entry, deletion, assignment, release, acquisition, and cancellation of the control right.

Parameters serving as arguments of event names as the conference events include toolNumber, toolID, confID, userID, and floorID. The parameter toolNumer corresponds to an identification number corresponding to a type of common application, the parameter toolID corresponds to an identification number when using a plurality of common applications for one conference under different states, the parameter confID shows a conference room number (e.g., serial number), the parameter userID corresponds to a user's identification number, and the parameter floorID corresponds to an identification number corresponding to a type of control right.

As shown in FIG. **3**, the conference control section 42_i comprises a conference control request processing section 52_i, a conference event processing section 54_i, a common application registration section **56***i*, a conference event distribution section 58_i, a conference event reception section 60_i, a common application table **62***i*, a callback routine table 64_i, and a conference table 66_i.

The conference control request processing section 52_i processes a request for conference state change from the terminal of its own or another terminal. The conference event processing section 54_i generates or delivers a conference event corresponding to a conference state change. The common application registration section 56_i dynamically registers a common application at start or the like. The conference event distribution section 58_i distributes and outputs a conference event for a conference state change request to enrollees who can attend the conference. The conference event reception section 60_i receives a conference event for a conference state change request from another terminal.

The common application table 62_i is provided with serial numbers for common applications and stores serial numbers, common application names, the address of the module concerned where one common application is loaded in RAM (memory) as a module, and the address of the common application initialization routine. The callback routine table 64_i stores common application serial numbers in the common application table 62_i, event names, and call back routine address. The conference table 66_i stores the number for a conference room which is currently opened and in which the user 18_i can participate, the conference room name, and the list of users who can use the conference room. Memory in which common applications are loaded is not restricted to RAM. It is possible to separately connect memory so that data is transferred to and from a terminal or add EMS memory or a cache memory. In this case, it is also possible to form main memory by adding EMS memory or cache memory to the terminal so that data can be loaded. This constitution is preferable because the device constitution is simplified.

A common application name (e.g., name of a dynamic link library to be loaded) used by a user in the conference room system **11** is stored beforehand in the common application management section 44_i as an application file APF.

The following is a description of the functions of this embodiment. For each terminal described below, a conference room to be opened (used) when starting a conference program references the conference table 66_i. That is, a user who can attend a plurality of conference rooms is repeatedly entered in the conference table 66_i. Therefore, for example, one or more conference rooms are selected by displaying the conference room list in the conference table 66_i or a plurality of conference rooms is selected by default at the start of the program. One field is registered in the conference table 66_i when a conference device is held. For example, when the user 18_i opens a conference room, an identification number is given to the conference room and a conference room name corresponding to the identification number and a list of users who can attend the conference room are stored. It is preferable to obtain consent from persons who can attend the conference room before storing a list of them. Though a plurality of conference rooms can be registered in the conference table 66_i, the use of a single conference room use is described below to simplify the description. Executing a plurality of conference rooms on a terminal is to process a plurality of conference rooms by means of individual time-sharing one by one. Therefore, this processing corresponds to the processing of each conference room by means of individual time-sharing interrupts or so-called window processing displaying a plurality of conference rooms on a screen and selectively using a specific conference room.

First, a method for obtaining only the common application necessary for operating the conference concerned or a common application to be executed only by each user from a plurality of common applications included in the common application section 50_i and how the conference control section of each user transfers data to and from a common application are described below together with "Starting routine" in FIG. **5**.

When a power switch (not shown) of a PC 16_i of a terminal is turned on and the terminal 16_i is ready for operation and the conference room system **11** for performing operations such as opening of a conference and participation in the conference is started through a keyboard, the common application registration section **56*i*** of the conference control section 66_i reads one common application name from the application file APF storing common application names (step **102**). Then, it is determined whether all common application names stored in the application file APF are read by determining the end of the application file APF (step **104**). In the case of a negative determination, a module stored in memory **34*i*** corresponding to a common application with the read name is loaded in RAM (memory) by the common application registeration section 56_i (step **106**). The common application registration section 56_i gives a serial number to the common application with the read name and enters the serial number, the common application name, the address of the module loaded in RAM (memory), and the address of an initialization routine in the common application table 62_i (step **108**). The initialization routine is located at a predetermined location (e.g., first opening function) of the loaded module, which sets a default and includes the address of a callback routine corresponding to a conference event included in a common application. Then, the common application registration section 56_i calls the initialization routine of the common application with the read name (step **110**) and obtains the address of a callback routine corresponding to a conference event from the called initialization routine (step **112**). Then, the common application registration section 56_i establishes a correspondence between the address of the callback routine thus obtained, a common application number and an event name and enters the common application number, event name, and callback routine address in the callback routine table 64_i (step **114**).

The above processing is repeated for all common application names stored in the application file APF.

Therefore, the conference control section 42_i defines conference state change details as conference events (FIG. **4**) and reports defined conference events to the common application section 44_i. Thereby, it is possible for a plurality of conference events of a common application to keep a consistent state independent of the conference state change. They correspond to window events in a window system. Each common application registers its own callback routine in the conference control section 42_i for a conference event required by each common application to keep conference event consistency for each terminal in the initialization routine of each common application. When conference states change, the conference control section 42_i calls the callback routine registered for the conference event. Thereby, it is possible to dynamically incorporate common applications, which are for a conference necessary at the start of a conference program without changing the states of the conference control section.

A method for sharing a common application currently executed in the terminal of a user by all conference enrollees in the same state independent of a conference room currently opened i s described below by referencing FIGS. **6** to **9**. To simplify the description, it is assumed that a common editor S which is the common editor S serving as a common application locally executed only at the terminal 16_A of a user A in a conference room opened by three users A, B, and C is shared by the three users. It is also possible to assume that a common application executed only by one user outside the conference room is shared by three users.

According to input through a keyboard 26_A of the user A, a request for sharing the conference event of the common editor S is output to a conference control section 42_A of the terminal 16_A of the user A (step **202**). This request is input to a conference control request processing section 52_A (signal path G1, FIG. **7**).

The conference control request processing section 52_A of the user A generates a conference event CE with the event name BEGINSHARE necessary for processing and outputs the conference event (event name: BEGINSHARE) to a conference event distribution section 58_A (step **204**, signal path G2). The conference event distribution section 58_A references a conference table 66_A (signal path G3 in FIG. **7**) to determine that users A, B, and C are enrollees at a conference room and distributes the conference event to these three users (step **206**). The conference event is also distributed to the user A who requested sharing of the conference event and the transfer data designating users B and C is output to terminal 16_B and 16_C of users B and C through a communication control section 40_A and the network **12** (signal path G4).

Conference event reception sections 60_A, 60_B, and 60_C of users A, B, and C receive a conference event. Details of processing for users B and C are described later. The conference event received by the conference event reception section 60_A is output to a conference event processing section 54_A (step **208**, signal path G5).

The conference event processing section 54_A reads a conference number and common application number from the parameter of the input conference event(depicted in FIG. **4**), references a callback routine table 64_A (signal path G6) to obtain the address of a callback routine corresponding to the conference event of the common application, and requests starting of the callback routine (signal path G7). A common application management section 44_A calls the requested callback routine and starts it (step 210). The conference event processing section 54_A waits for the next processing until the processing of the callback routine terminates. This processing is executed in the callback routine of the common editor independently of the conference control section.

The callback routine of the common application orders the common editor S of the user A to terminate processing (signal path G8) and the common editor S terminates processing currently being executed (step 212).

When receiving termination of processing, the common application section of the user A stores the current state of the common editor S (e.g., read data file, portion currently edited, or cursor location) and outputs the stored state to users B and C (step 214, signal path G9).

After state output to users B and C terminates, the common application section 50_A outputs a termination signal to the conference event processing section 54_A to terminate processing of conference event which is conference control in the conference event processing section 54_A (step 216, signal path G10).

Processing by terminals 16_B and 16_C of users B and C is described below. Because almost the same processing is performed for users B and C, processing by the terminal 16_B of the user B is described below for simplicity.

When a signal based on transfer data is input to the conference event reception section 60_B from the terminal of another user (signal path G20, FIG. 9), the sharing routine in FIG. 8 is executed and the conference event reception section 60_B receives a conference event. The conference event received by the conference event reception section 60_B is output to the conference event processing section 54_B (step 222, signal path G21).

The conference event processing section 54_B reads a conference number and a common application number from the parameter of the input conference event and references the conference table 66_B (signal path G22*a*) and the callback routine table 64_B (signal path G22) to obtain the address of a callback routine corresponding to the conference event of a conference room in which a common application is included and requests starting the callback routine (signal path G23). A common application managing section 44_B calls the requested callback routine to start it (step 224). The conference event processing section 54_B waits for the next processing until processing of the callback routine terminates.

The callback routine of a common application designates starting the common editor S serving as a common application corresponding to the common editor S of the user A (signal path G24) and starts the common editor S at the terminal B of the user B (step 226). In this case, it is preferable for the common application to store all current states or accept a state sent from another unit as a new state to realize sharing during execution.

When receiving termination of a start, the common application section of the user B waits until reception of transfer data to be shared sent from another user terminates (step 228, signal path G25). When reception of transfer data terminates, the common editor of the user B performs processing so as to coincide with the state of the common editor S of the user A (step 230). After processing terminates, the common application section 50_B outputs a termination signal to the conference event processing section 54_B to terminate processing of the conference event which is conference control in the conference event processing section 54_B of the user B (step 232, signal path G26).

Synthetically, a request for sharing the conference event of the common editor is issued to the conference control section of the user A from the user A, the conference control section of the user A communicates with conference control sections of users B and C, and a callback routing for sharing the conference event of the common editor S is called by terminals of users A, B, and C. The callback routine of the common editor of the user A communicates with the callback routine of the common editor of users B and C. The common editor of the terminal of the user A first terminates execution of all processing before sharing to wait for the state to become steady and thereafter starts the common editor with terminals of users B and C. After it is confirmed that the common editor is started by editors of users B and C, the current state of the common editor S is sent to users B and C. When the above processing terminates, users A, B, and C terminate callback routine processing. When callback routine processing terminates and control returns to the conference control section, common editors S in the same state are working in terminals of three users. Thereafter, it is possible to input data to the common editor.

Thus, it is possible to share a common application under execution by clearly separating conference control section processing from that of the common application, managing the conference event notification of sharing a common application by the conference control section, providing the common application with functions for storing the current execution state and realizing a new state, and shifting to the same state by synchronizing common terminal applications.

The second embodiment is described below. In the first embodiment, a case is described in which only the terminal of each user participating in a conference is connected by a network or the like. However, the present invention is not restricted to the above case. As shown in FIG. 10, it is also possible to manage a conference by connecting one or more centralized managing devices 17 to a network or the like. For the second embodiment, the same portion with that of the first embodiment is provided with the same symbol and its detailed description is omitted. Therefore, only portions different from those of the first embodiment are described below. The conference system 10 of this embodiment, as shown in FIG. 10, comprises an integrated controller 17 comprising a host computer such as a microcomputer connected to a network 12 and L (L is a natural number) terminal 16_1 to 16_L.

As shown in FIG. 11, the integrated controller 17 comprises a communication control section 41 for communicating data with a terminal and a conference control section 43 for controlling conference operations such as activation and termination of a conference room opened among a plurality of terminals and enrollees entering and exiting the conference. The communication control section 41 is connected to the network 12 and also to the conference control section 43. The conference control section 43 comprises a conference control request processing section 53, a conference event distribution section 59, and a data storage section 51 for holding an executed state of a common application. The conference control request processing section 52_i generates a conference event corresponding to a change in the conference state when a request for changing conference states is output from various terminals. The conference event distribution section 58_i distributes and outputs a conference event for a conference state change request to enrollees in the conference.

A terminal 16_A comprises a PC 22_A, a display 24_A and a keyboard 26_A like that of the first embodiment. Because other terminals have the same constitution, their description is omitted. As shown in FIG. 12, the PC 22_A of the terminal 16_A mainly comprises a communication control section 40_A for communicating data with another PC_j ($j:1\leqq j\leqq L$, $i\neq j$) and the integrated controller **17**, a conference control section 42_A (described later) for executing a conference room opened at a terminal, a common application section 50_A comprising one or more programs shared by users and concurrently used in a conference room, and a common application management section 44_A. The communication control section 40_A is connected to the network **12** and a keyboard 26_A and also connected to the conference control section 42_A and the common application section 50_A. The communication control section may perform one-to-one data transfer to and from other terminals separately from an opened conference. The conference control section 42_A is connected to the common application section 50_A and common application management section 44_A. The common application section 50_A is connected to the common application management section 44_A.

The conference control section 42_A comprises a common application registration section 56_A a conference event distribution section 58_A, a conference event reception section 60_A , a common application table 62_A, a callback routine table 64_A, and a conference table 66_A.

The conference event processing section 54_i delivers a conference event corresponding to a conference state change. The common application registration section 56_i dynamically registers a common application at starting or the like. The conference event reception section 60_i receives a conference event for a conference state change request sent from the integrated controller.

The functions of this embodiment are described below. For each terminal of this embodiment, a conference room opened (used) by starting a conference program is performed by referencing a conference event and the conference table 66_i from the integrated controller. That is, a user who can use a plurality of conference rooms is repeatedly entered in the conference table 66_i. Therefore, one or more conference rooms are selected from the conference room list of the conference table 66_i by the integrated controller or one or more conference rooms are selected by default, for example, at the start of the program. A field is registered in the conference table 66_i and the table 66_i transmits the registered field to the integrated controller **17** when a conference is opened. Because it is possible to register a plurality of conference rooms in the conference table 66_i, a case of registering one conference room is described below for simplicity.

First, the start processing of this embodiment is described below. Because the start of this embodiment is almost the same as that of the first embodiment in FIG. **5**, it is described by referencing FIG. **5** by giving the same number to the processing equivalent to that in FIG. **5**. When a power switch (not shown) of a terminal PC 16_i is turned on, the terminal 16_i is ready for operation and the conference room system **11** for performing operations such as opening of a conference and participation in the conference is started by a designation through a keyboard, and data is sent to an integrated controller. The integrated controller references a conference table **67** and sends a conference event showing the start to the started terminal. The common application registration section 56_i of the conference control section 66_i of the terminal 16_i receiving the conference event showing the start reads a common application name from the application file APF storing common application names (corresponding to step **102**, FIG. **5**). Then, it is determined whether all common application names stored in the application file APF are read by determining the end of the application file APF (step **104**). In the case of a negative determination, a module stored in memory 34_i corresponding to a common application with the read name is loaded in RAM (memory) by the common application registration section 56_i (step **106**). The common application registration section 56_i gives a serial number to the common application with the read name and enters the serial number, the common application name, the address of the module loaded in RAM (memory), and the address of an initialization routine in the common application table 62_i (step **108**). The initialization routine is located at a predetermined location (e.g., first opening function) of the loaded module, which shows the type of conference and a default and includes the address of a callback routine corresponding to a conference event included in a common application. Then, the common application registration section 56_i calls the initialization routine of the common application with the read name (step **110**) and obtains the address of a callback routine corresponding to a conference event from the called initialization routine (step **112**). Then, the common application registration section 56_i establishes a correspondence between the address of the callback routine thus obtained and to a common application number and an event name and enters the common application number, event name, and callback routine address in the callback routine table 64_i (step **114**).

The above processing is repeated for all common application names stored in the application file APF.

Therefore, the conference control section 42_i defines conference state change details as conference events (FIG. **4**) and reports defined conference events to the common application section 44_i. Thereby, it is possible for a plurality of conference events of a common application to keep a consistent state independent of the conference state change.

A procedure for sharing a common application under execution is described below by referencing FIGS. **11** to **13**. Hereafter, to simplify the description, it is assumed that a common editor S which is the common editor S serving as a common application locally executed only at the terminal 16_A of a user A in a conference room opened by three users A, B, and C is shared by the three users. It is also possible to assume that a common application executed only by one user outside the conference room is shared by three users.

According to input through a keyboard 26_A of the user A, a request for sharing the conference event of the common editor S is output to the integrated controller **17** (signal path G32 in FIG. **11** and the signal path **31** in FIG. **12**).

The conference control request processing section **53** of the integrated controller **17** generates an conference event CE with the event name BEGINSHARE necessary for processing and outputs the conference event (event name: BEGINSHARE) to a conference event distribution section **59** (signal path G33, FIG. **11**). The conference event distribution section **59** references a conference table **67** (signal path G34 FIG. **11**) to determine that users A, B. and C are enrollees at a conference room and distributes the conference event to these three users. The conference event is also distributed to the user A who requests sharing of the conference event and terminals 16_B and 16_C of users B and C through a communication control section **41** of the integrated controller **17**, network **12**, and communication control sections 40_A, 40_B, and 40_C (signal path G35, FIG. **11**; signal path **36**, FIG. **12**; signal path **37**, FIG. **13**).

Conference event reception sections 60_A, 60_B, and 60_C of users A, B, and C receive a conference event. The conference event received by the conference event reception sections is outputted to each conference event processing section (signal path 38, FIG. 12; signal path 39, FIG. 13). Each conference event processing section **54** reads a conference number and common application number from the parameter of the input conference event, references a conference table and a callback routine table (signal path 40, FIG. 12; signal path 41, FIG. 13) and obtains the address of a callback routine corresponding to the conference event of the common application to request starting the callback routine (signal path 42, FIG. 12; signal path 42*a*, FIG. 13). Each common application management section calls and starts the requested callback routine. Each conference event processing section waits for the next processing until the processing of the callback routine terminates. Subsequent processing is executed in the callback routine of a common editor independent of the conference control section.

The callback routine of the common application of the user A requesting sharing orders the common editor S of the user A to terminate processing (signal path G43) and terminates the processing of the common editor S currently being executed. The common application section of the user A receives the termination of processing, stores the current state of the common editor S, and outputs the stored state to the integrated controller **17** (signal paths G44 and G44*a*). Thereafter, to terminate the processing of the conference event which is the conference control in the conference event processing section **54**$_A$, the common application section **50**$_A$ outputs a termination signal to the conference event processing section **54**$_A$ (signal path G45).

Then, processing by terminals **16**$_B$ and **16**$_C$ of users B and C for which sharing is requested is described below. Because almost the same processing is performed for users B and C, processing by the terminal **16**$_B$ of the user B is described below for simplicity.

Base on the conference event sent from the integrated controller **17**, the conference event processing section **54**$_B$ reads a conference number and a common application number from the parameter of the input conference event, references a conference table and the callback routine table **64**$_B$ and obtains the address of a callback routine corresponding to the conference event of the conference room including a common application to request starting the callback routine (signal path G42*a*). The common application managing section **44**$_B$ calls and starts the requested callback routine. The conference event processing section **54**$_B$ waits for the next processing until callback routine processing terminates.

The callback routine of a common application orders starting the requested common editor S (signal path G46) and starts the common editor S at the terminal **16**$_B$ of the user B. In this case, it is preferable to set the common application so that it can store all current states and receive a state sent from an external unit as a new state to realize sharing during execution.

By receiving the start termination, the common application section of the user B waits until the reception of transfer data from the integrated controller **17** terminates (signal path G47). The user B then processes the common editor S so that the state of the common editor B coincides with the state of the common editor S of the user A. After processing terminates, the common application section **50**$_B$ outputs a termination signal to the conference event processing section **54**$_B$ to terminate processing of the conference event which is conference control in the conference event processing section **54**$_B$ of the user B (signal path G48).

Therefore, the conference control section of each user communicates with the integrated controller, the callback routine for sharing the conference event of the common editor S by terminals of users A, B, and C, and it is possible to perform processing so that common editors in the same state are operated in terminals of the three users.

Thus, because this embodiment manages generation and reporting of a conference event sharing a common application by an integrated controller, it is possible to decrease additional processing of each terminal and share a common application under execution by using fewer execution programs and less memory.

The conference system **10** of the above embodiment also makes it possible to send or receive voice information including the voice of a user and music and animation information including the face of the user as transfer data. In this case, it is also possible to send or receive voice and animation information through transmission means **13** such as a separate line or separate network as shown in FIG. **14**. Therefore, by sending or receiving voice information and animation information through a separate line or network, it is possible to decrease data capacity and manage a conference in real time even using conference system having voice and animation information as data.

In the above embodiment, a case of using the annular network **12** is described. However, as shown in FIG. **15**, it is also possible to connect a plurality of terminal to one terminal **16**$_x$ (X=any one of 1 to N) or connect a plurality of terminals to an integrated controller **17**.

As processing for making other user attend an opened conference, it is preferable to perform attendance request processing for asking whether the user attends the conference with a terminal **16**$_j$ for which attendance is requested. In this case, it is preferable to set a terminal **16**$_i$ of the attendance request side when requesting another user to attend the conference so that the terminal cancels the above attendance procedure when transfer data showing that other user **18**$_j$ refuses attendance is returned or no response returns because the other user **18**$_j$ does not operate the terminal **16**$_j$ or is out (after a preset time of a timer lapses).

Moreover, to proceed with the conference, the common application use right may come into contention. In the case of the above common blackboard, the common blackboard use right contends because a plurality of users simultaneously request the common blackboard use right. In this case, it is necessary to make users wait their turn to obtain the use right by giving priority to the use right. Data showing use right permission and the turn are sent to the conference event reception section and the common application section of each user. Based on this data, the conference event processing section allows a user to use the common blackboard. It is preferable to set up an application program for the common blackboard to enable a user to input data with an input unit such as a keyboard or mouse and to send data for making input by users other than the permitted user impossible to prevent erroneous input.

As described above, the present invention has advantages that only a dynamically necessary application among a plurality of applications can be incorporated into a real-time conference system for multiple enrollees in which applications are executed by the conference device of each enrollee at the start of the conference system and these applications under execution can be started and shared by the terminal of another user in the same state.

Having thus described our invention, what we claim as new and desire to secure by Letters Patent is:

1. In a conference system including a plurality of conference devices with at least one requesting conference device

executing an application program to be used in a conference, a method for dynamically sharing the application program among the plurality of conference devices while maintaining a same state of the application program at each of the plurality of conference devices, said method comprising the steps of:

communicating a request to share the application program from said requesting conference device executing the application program to other ones of said conference devices enrolled in the conference;

storing a current state of the application program in the requesting conference device;

loading the application program by said other ones of said conference devices enrolled in the conference;

communicating the current state from the requesting conference device to said other ones of said conference devices enrolled in said conference; and

processing the application program by said other ones of said conference devices enrolled in said conference to acquire a state equal to the current state.

2. The conference system as recited in claim 1 wherein the application program is a blackboard program for displaying a screen on said conference devices on which users can write messages.

3. In a conference system including a plurality of conference devices wherein a requesting conference device is executing an application program, said requesting conference device adapted for dynamically sharing the application program among the plurality of conference devices while maintaining said plurality of conference devices in a same state, said requesting conference device comprising:

application program storage means for storing an application program used for a conference;

callback protocol storage means, coupled to the application program storage means, for storing a processing program along with identifiers for sharing the application program;

conference control means for maintaining all conference devices in a same state coupled to the callback protocol storage means, wherein one or more registered identifiers of the processing program corresponding to event information indicating a changed state of a conference are previously stored, said conference control means communicating the event information corresponding to the changed state to all said conference devices, said conference control means further for changing a conference execution state to be equal to the changed state corresponding to communicated event information and executing the processing program corresponding to the communicated event information by referencing the registered identifiers; and

communication control means, coupled to the conference control means, for communicating transferred data including the event information.

4. The conference device of claim 3, where the plurality of conference devices communicatively couple so that the event information can be input or output.

5. The conference system as recited in claim 3 wherein the application program is a blackboard program for displaying a screen on said conference devices on which users can write messages.

* * * * *

19

US005940082A

United States Patent [19]

Brinegar et al.

[11] **Patent Number: 5,940,082**

[45] **Date of Patent: Aug. 17, 1999**

[54] **SYSTEM AND METHOD FOR DISTRIBUTED COLLABORATIVE DRAWING**

[76] Inventors: **David Brinegar**, 1508 Oxford St., Berkeley, Calif. 94143-0450; **David Hingston**, 47 Eastwood Dr., San Francisco, Calif. 94112

[21] Appl. No.: **08/798,280**

[22] Filed: **Feb. 14, 1997**

[51] **Int. Cl.**^{6} **G06F 3/00**

[52] **U.S. Cl.** **345/442**

[58] **Field of Search** 345/440, 441, 345/442, 443

[56] **References Cited**

U.S. PATENT DOCUMENTS

5,206,934	4/1993	Naef, III	395/200
5,231,697	7/1993	Yamada	395/142
5,237,649	8/1993	Yamada	395/143
5,309,555	5/1994	Akins et al.	395/157
5,392,400	2/1995	Berkowitz et al.	395/200
5,425,109	6/1995	Saga et al.	382/187
5,442,788	8/1995	Bier	395/650
5,581,702	12/1996	McArdle et al.	395/200.04
5,608,872	3/1997	Schwatz et al.	395/200.04
5,717,879	2/1998	Moran et al.	395/339
5,726,669	3/1998	Obata et al.	345/2

OTHER PUBLICATIONS

Stults, "Experimental Uses of Video to Support Design Activities" (Palo Alto Research Center, Dec. 1988, copyright Xerox Corporation).

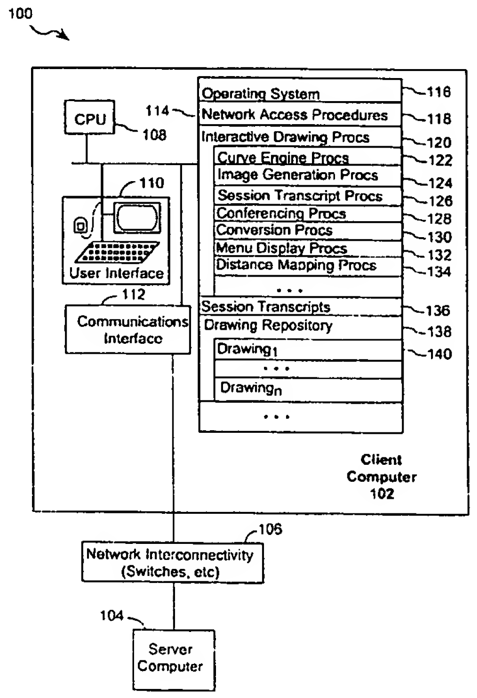
Tang, et al., "VideoDraw: A Video Interface for Collaborative Drawing" (System Sciences Laboratory, Palo Alto, California).

Primary Examiner—Phu K. Nguyen
Assistant Examiner—Cliff N. Vo
Attorney, Agent, or Firm—Gary S. Williams; Pennie & Edmonds LLP

[57] **ABSTRACT**

The present invention pertains to a system and method for managing a real-time distributed collaborative drawing activity. A community of collaborators associated with client computers are connected via a communications link with a server computer. Each collaborator contributes to the real-time design of a shared drawing that is displayed in each collaborator's virtual whiteboard. Contributions to the shared drawing as well as communications between the collaborators are transmitted to the server computer. The server computer broadcasts the communication to each of the collaborators engaged in the design activity. Each user has a set of drawing tools that are used to generate a variety of drawing strokes that edit the shared drawing. A curve generating procedure is used to capture the manner in which the user draws the stroke in order to emulate a selected drawing tool. A compact representation of the user's stroke is generated and transmitted to each collaborator.

27 Claims, 13 Drawing Sheets



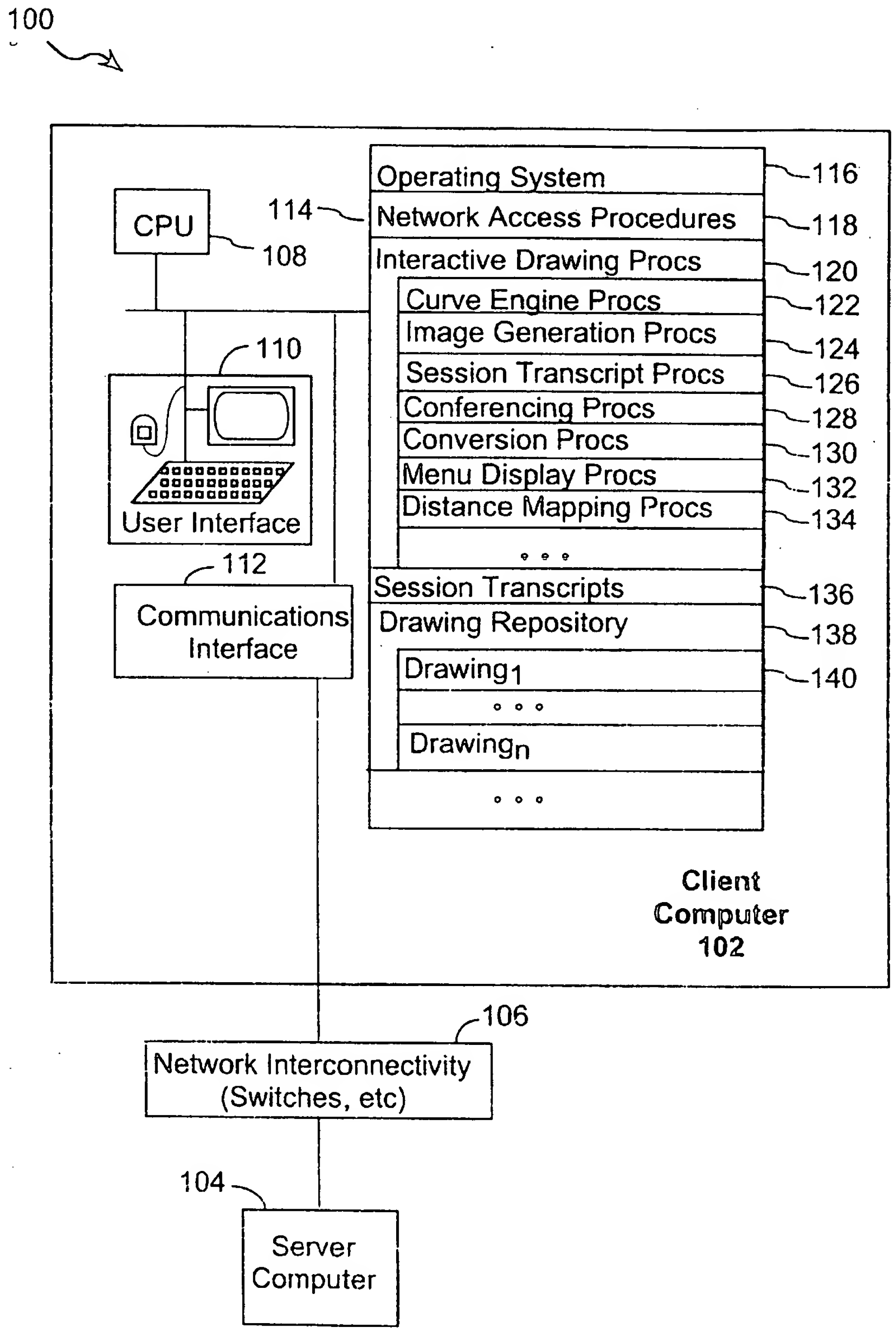

100
CPU
108
114
110
User Interface
112
Communications Interface
Operating System
116
Network Access Procedures
118
Interactive Drawing Procs
120
Curve Engine Procs
122
Image Generation Procs
124
Session Transcript Procs
126
Conferencing Procs
128
Conversion Procs
130
Menu Display Procs
132
Distance Mapping Procs
134
Session Transcripts
136
Drawing Repository
138
Drawing1
140
Drawingn
Client Computer 102
106
Network Interconnectivity (Switches, etc)
104
Server Computer

FIG. 1

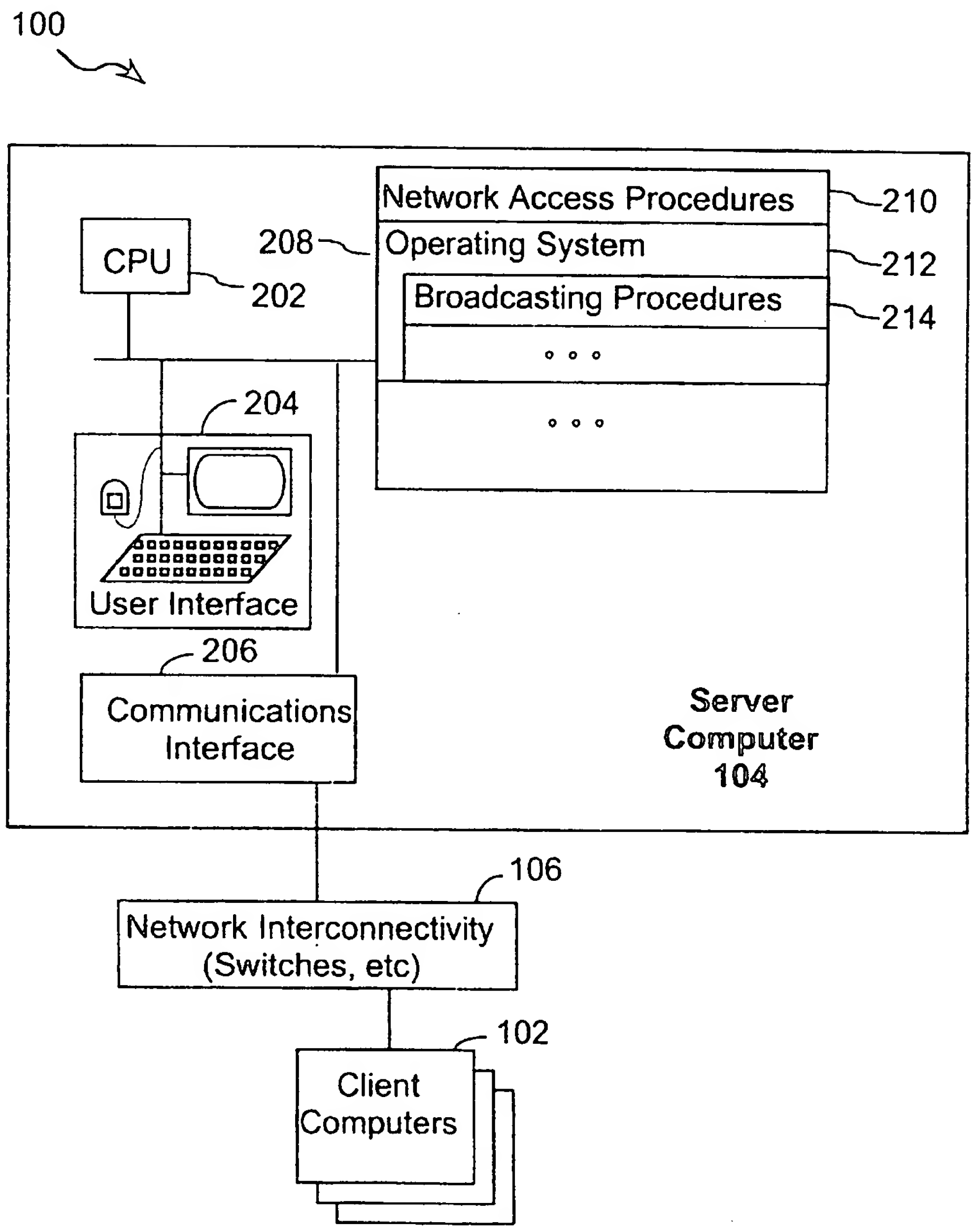

100
Network Access Procedures
210
208
Operating System
212
CPU
202
Broadcasting Procedures
214
204
User Interface
206
Communications Interface
Server Computer 104
106
Network Interconnectivity (Switches, etc)
102
Client Computers

FIG. 2

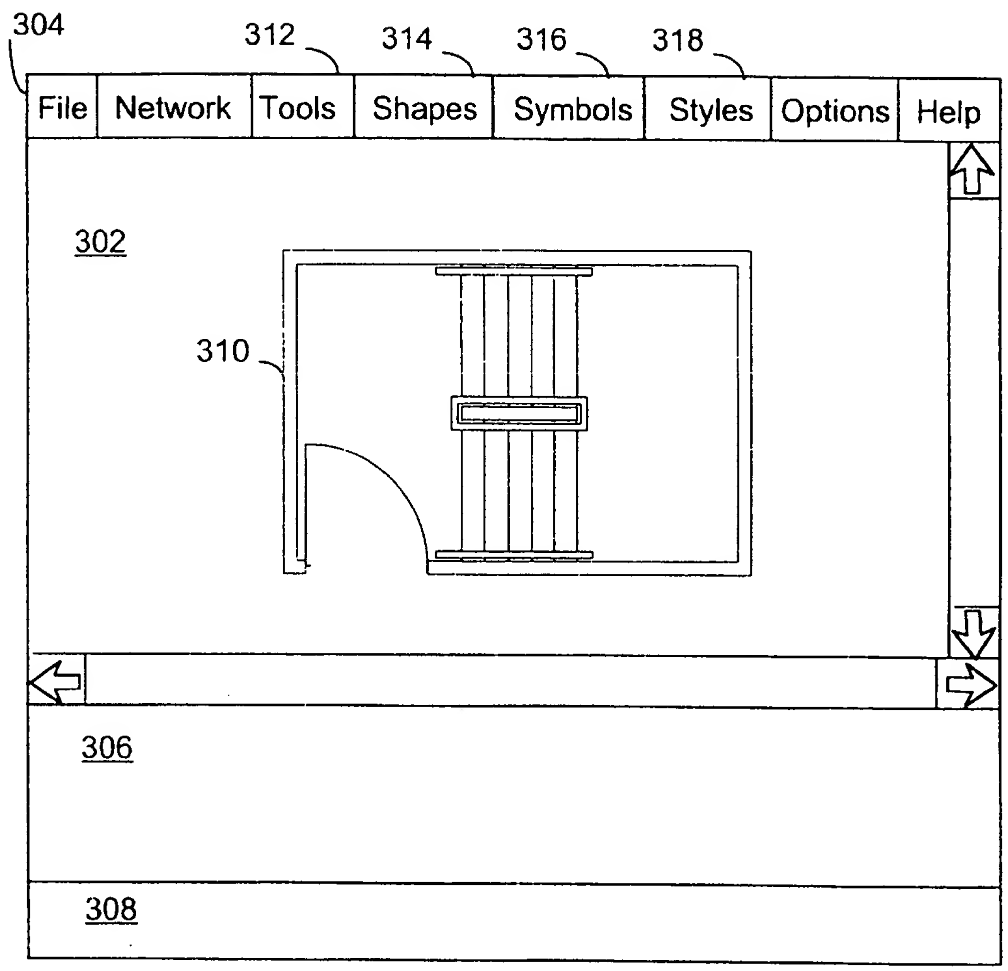

304
312
314
316
318
File
Network
Tools
Shapes
Symbols
Styles
Options
Help
302
310
306
308

FIG. 3

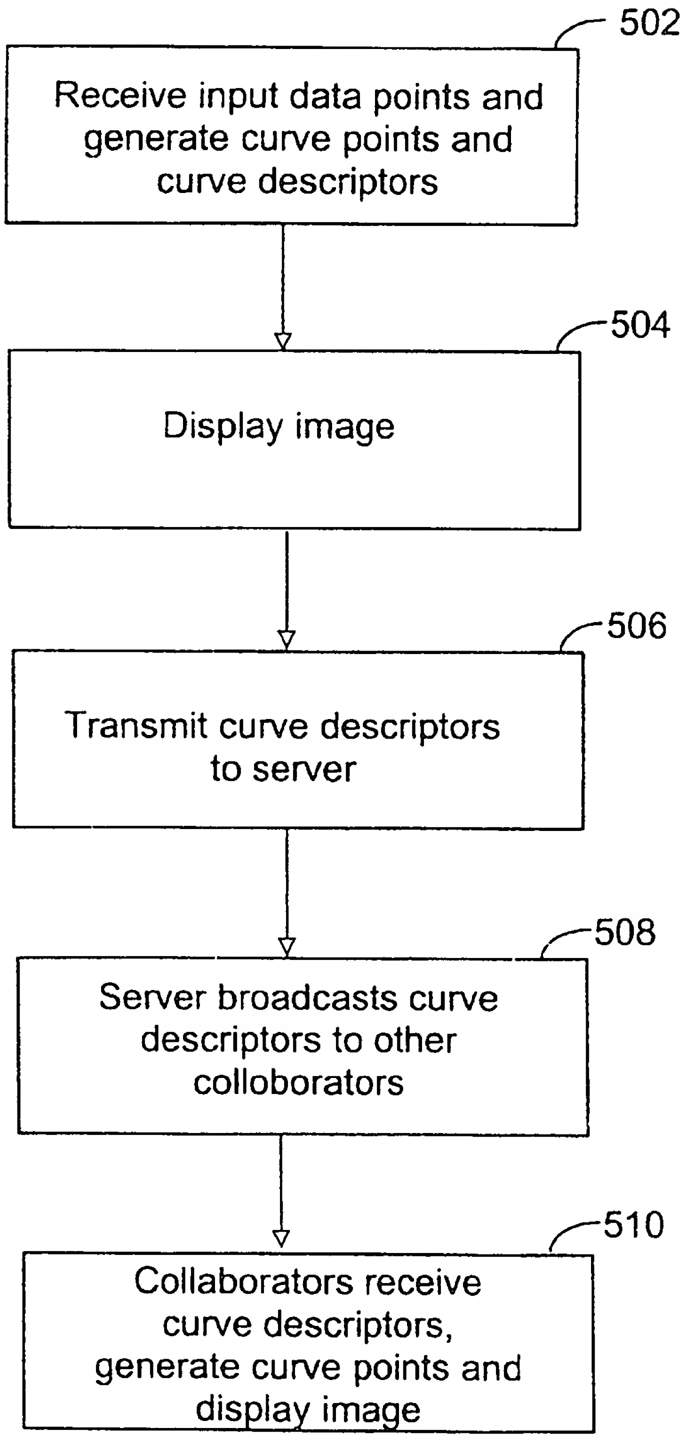

502
Receive input data points and generate curve points and curve descriptors
504
Display image
506
Transmit curve descriptors to server
508
Server broadcasts curve descriptors to other colloborators
510
Collaborators receive curve descriptors, generate curve points and display image

FIG. 5

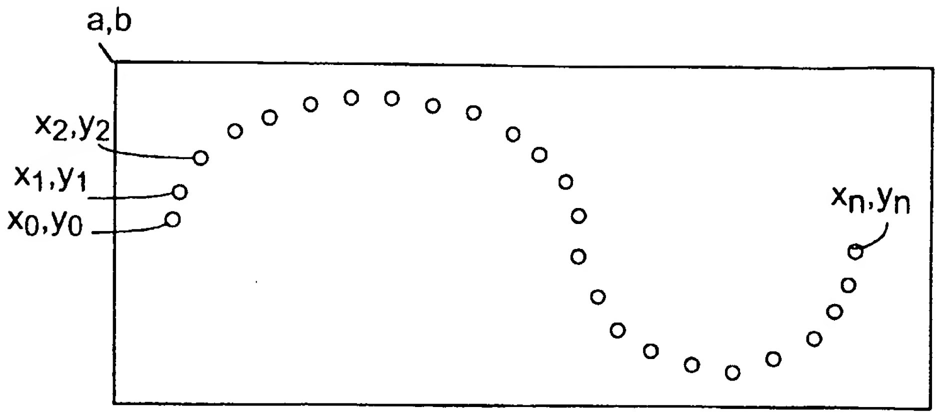


FIG. 6A

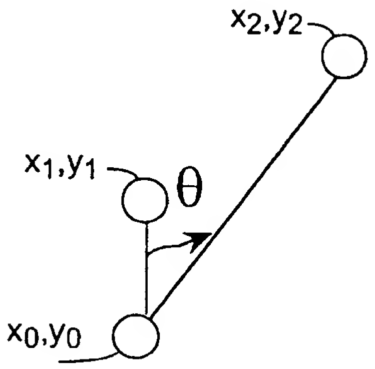


FIG. 6B

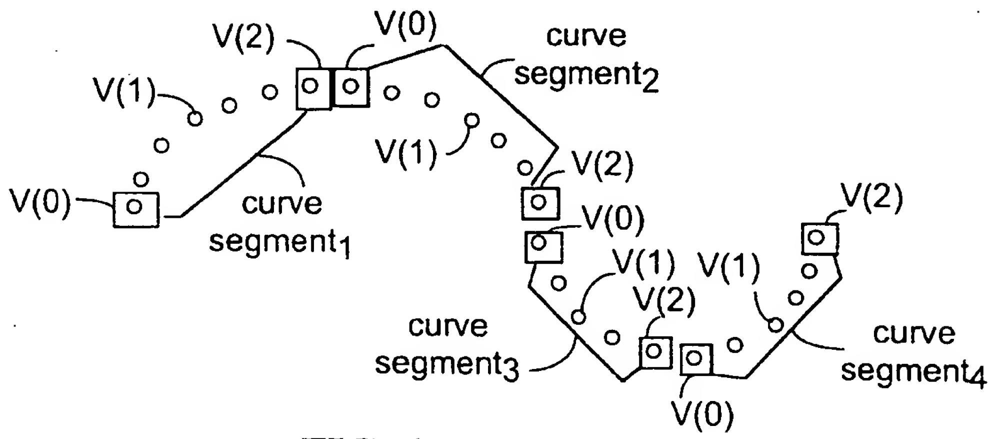


FIG. 6C

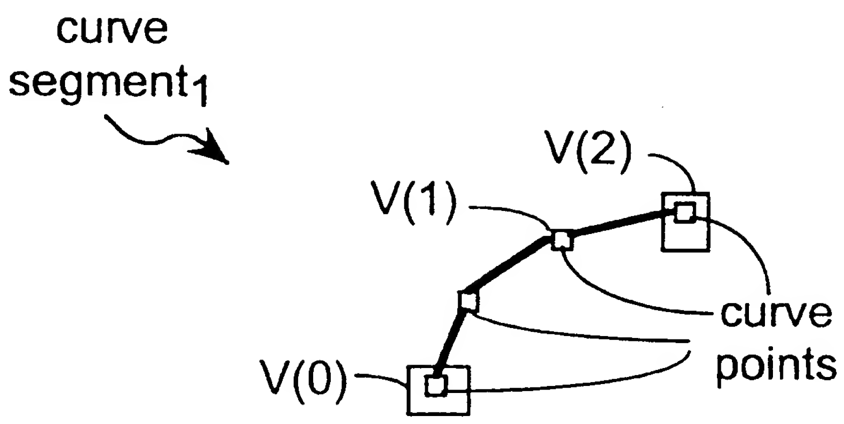


FIG. 6D

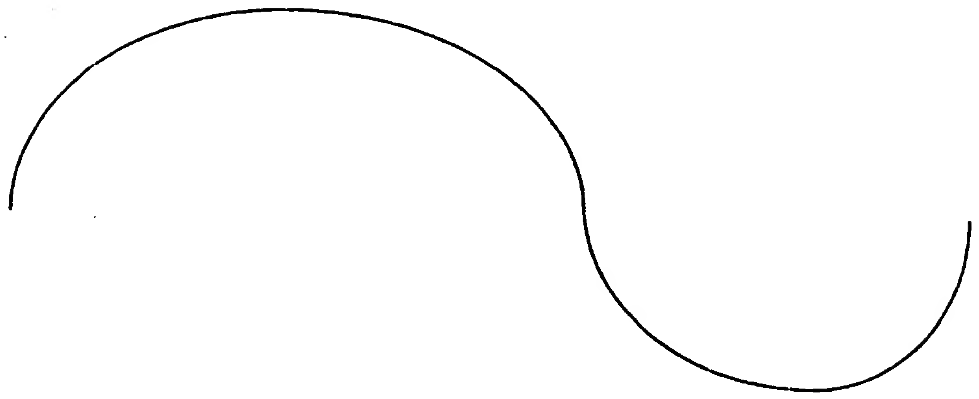


FIG. 6E

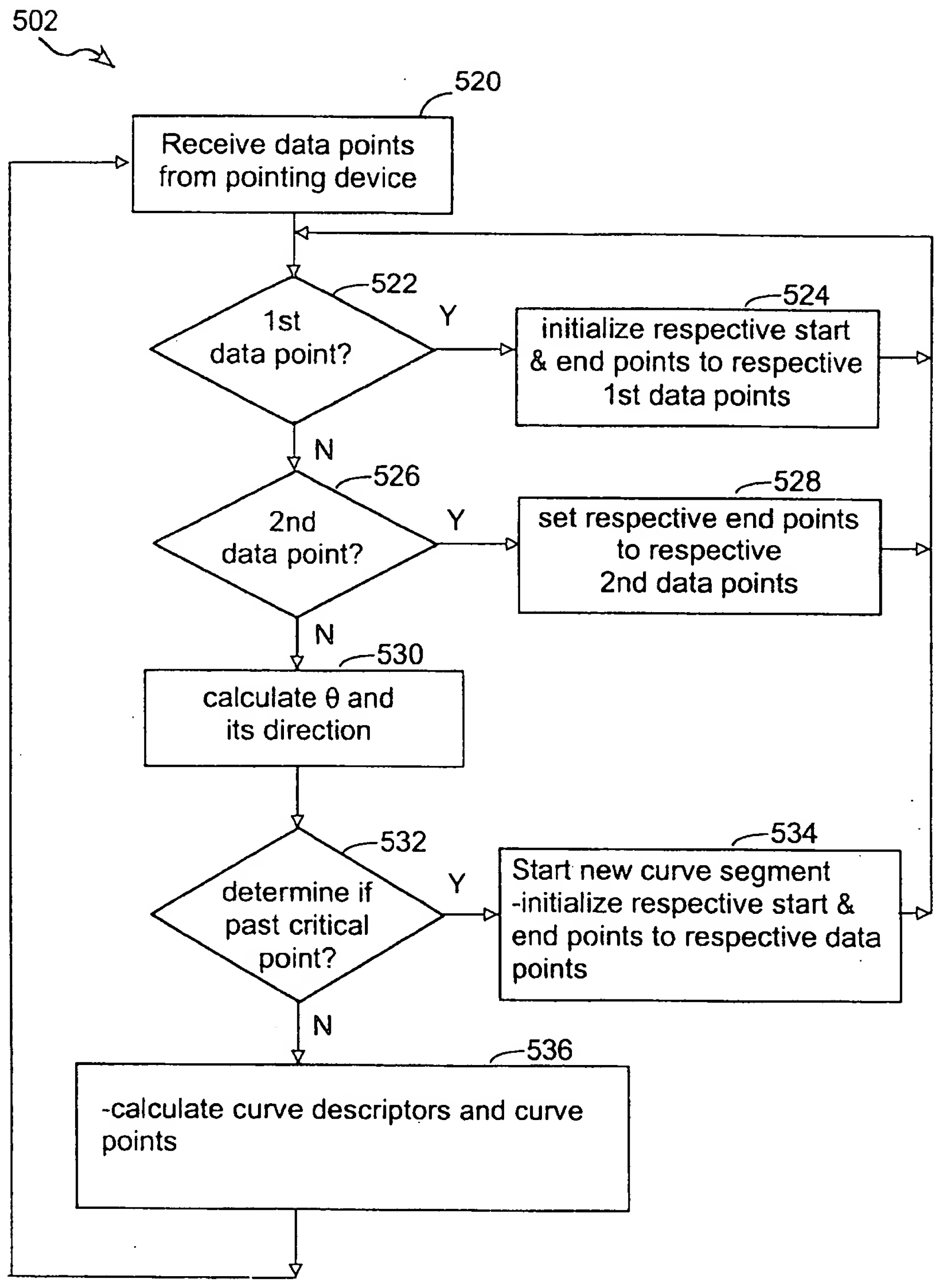

502
520
Receive data points from pointing device
522
1st data point?
Y
N
524
initialize respective start & end points to respective 1st data points
526
2nd data point?
Y
N
528
set respective end points to respective 2nd data points
530
calculate θ and its direction
532
determine if past critical point?
Y
N
534
Start new curve segment -initialize respective start & end points to respective data points
536
-calculate curve descriptors and curve points

FIG. 7

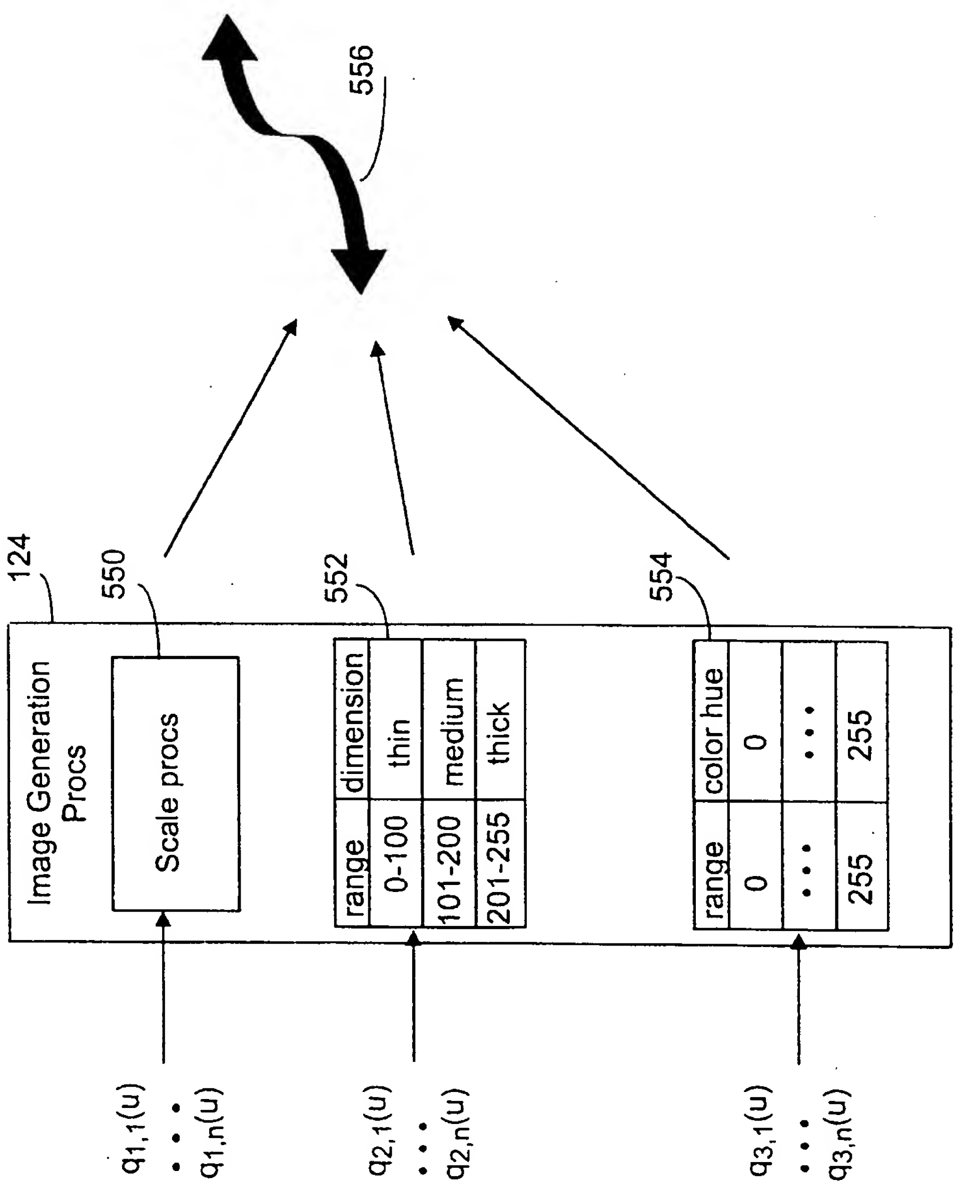

556
124
Image Generation Procs
550
Scale procs
552
range
dimension
0-100
thin
101-200
medium
201-255
thick
554
range
color hue
0
0
• • •
• • •
255
255
$q_{1,1}(u)$ • • • $q_{1,n}(u)$
$q_{2,1}(u)$ • • • $q_{2,n}(u)$
$q_{3,1}(u)$ • • • $q_{3,n}(u)$

FIG. 8

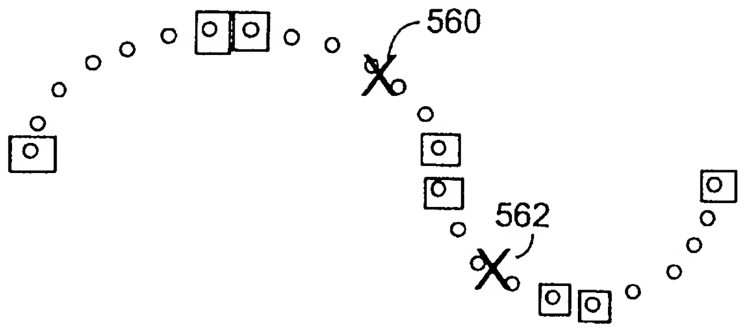


FIG. 9A

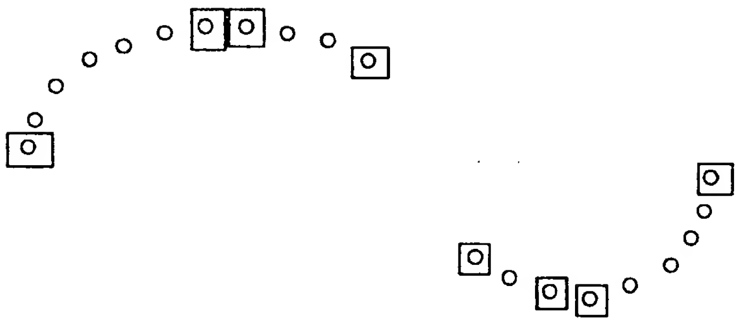


FIG. 9B

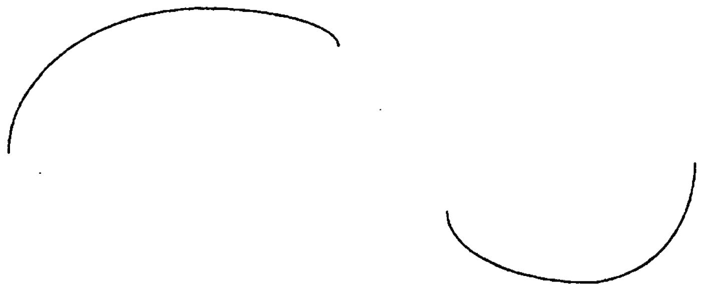


FIG. 9C

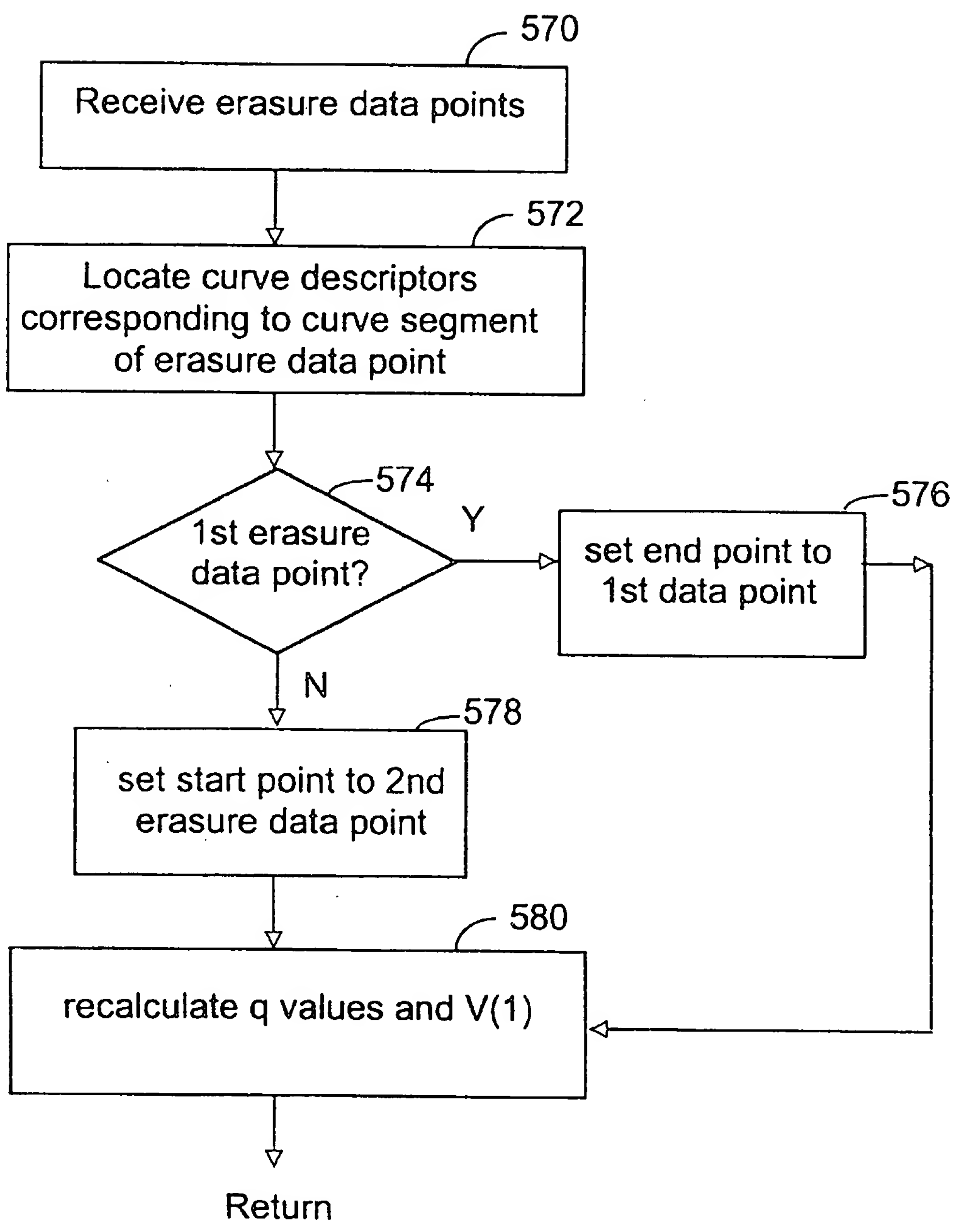

570
Receive erasure data points
572
Locate curve descriptors corresponding to curve segment of erasure data point
574
1st erasure data point?
Y
576
set end point to 1st data point
N
578
set start point to 2nd erasure data point
580
recalculate q values and V(1)
Return

FIG. 10

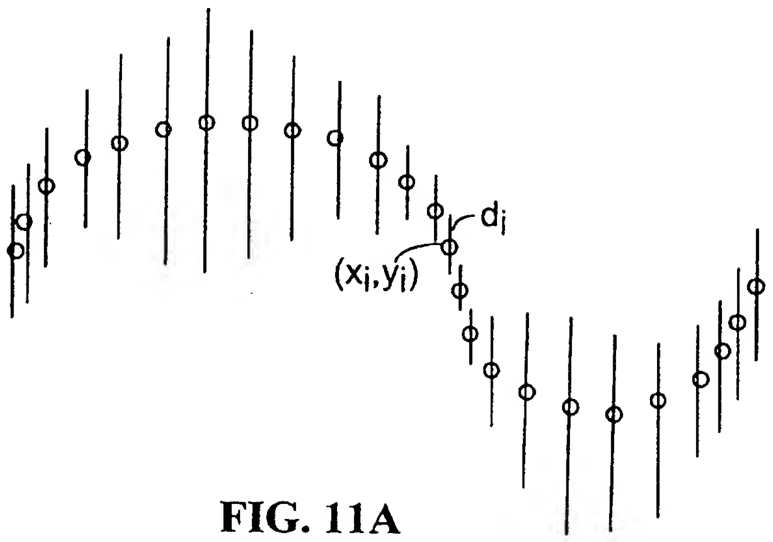


FIG. 11A

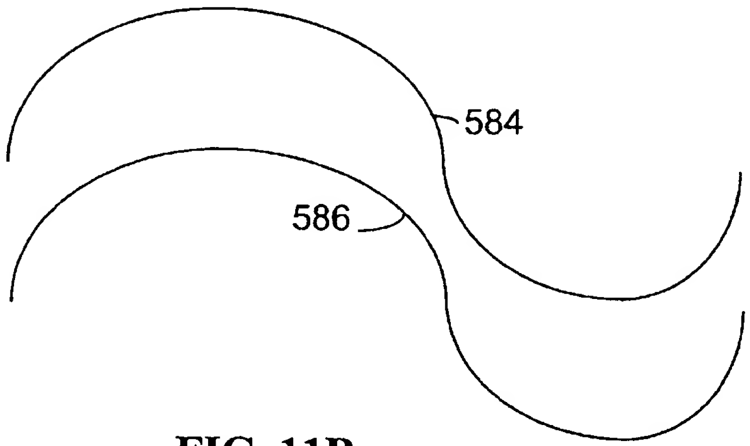


FIG. 11B

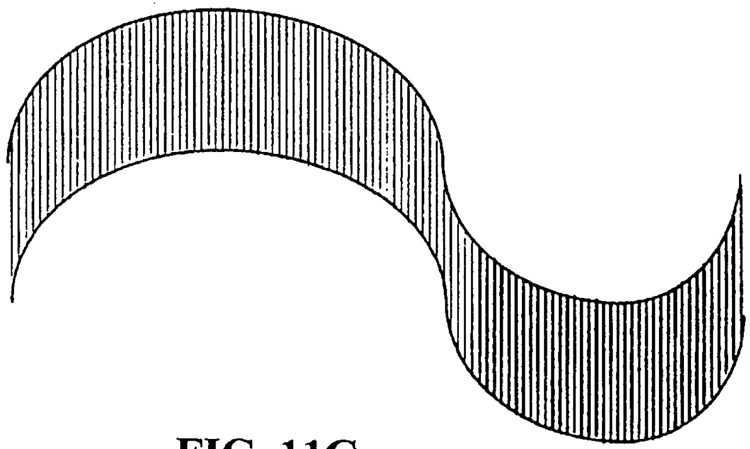


FIG. 11C

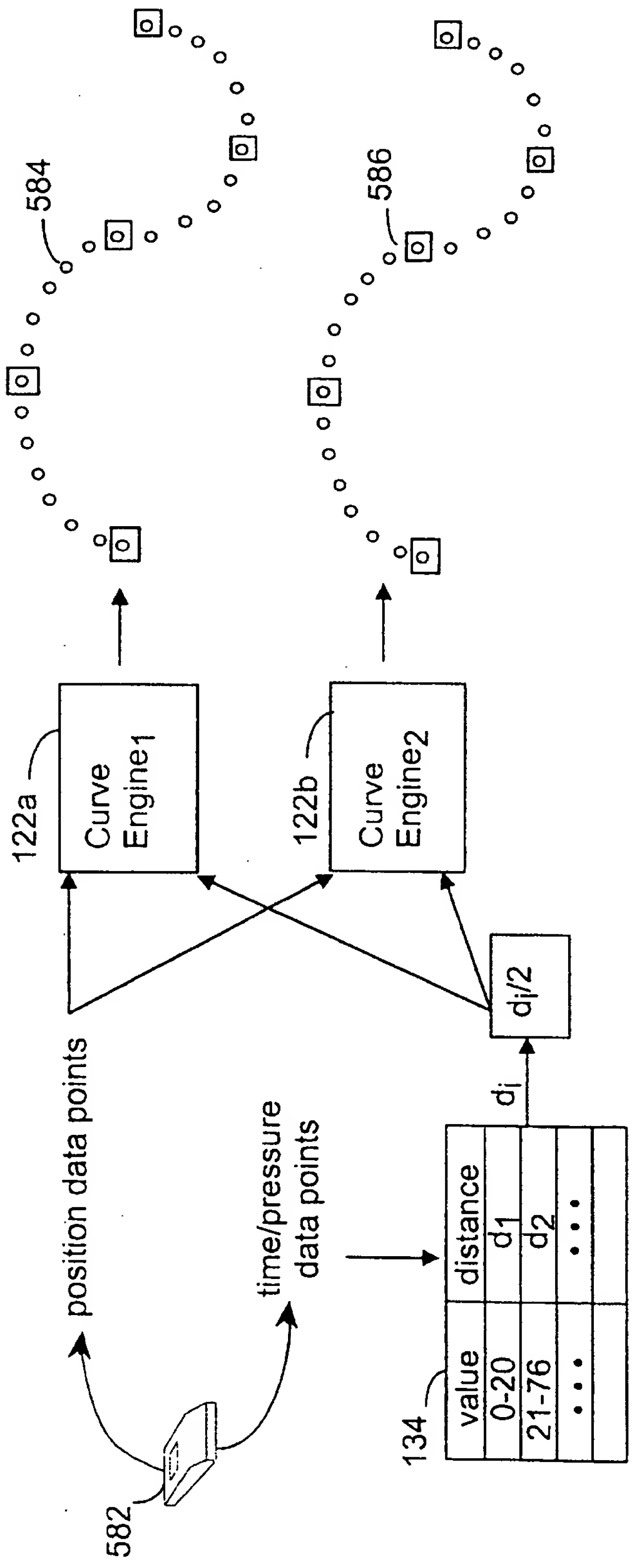

582
position data points
time/pressure data points
134
value
0-20
21-76
• • •
distance
d_1
d_2
• • •
d_i
$d_i/2$
122a
Curve Engine$_1$
122b
Curve Engine$_2$
584
586

FIG. 12

SYSTEM AND METHOD FOR DISTRIBUTED COLLABORATIVE DRAWING

The present invention relates generally to distributed computing systems and particularly to a system and method for managing a distributed collaborative design activity.

BACKGROUND OF THE INVENTION

Interactive conferencing systems provide users residing in disparate geographic locations the capability to participate in a real time design activity. Typically each user or participant is associated with a computer that is networked with the computers of the other participants. A display device associated with each user is used as a virtual whiteboard to display and receive drawings and other communications pertinent to the collaborative design activity. Each participant has the same copy of the shared drawing in its whiteboard. Each participant engages in the design activity by adding or editing the shared drawing. Alterations to the shared drawing are quickly transmitted to each participant in order to maintain a consistent state of the shared drawing amongst the participants. Thus, the success of such a system relies on an efficient methodology for quickly relaying graphical images to each of the participants.

SUMMARY OF THE INVENTION

The technology of the present invention pertains to an apparatus and method for supporting a real-time computer-networked collaborative design activity. The collaborative design activity includes a community of users that communicate with one another through client computers interconnected via a computer network such as the Internet. In an embodiment of the present invention, the collaborative design activity is a shared drawing activity between a community of users within the architecture engineering construction (AEC) industry. Each user utilizes a video display as a virtual whiteboard to engage in the collaborative design of a shared drawing.

Each user or collaborator in the collaborative drawing session has its own copy of the shared drawing. Each user can edit the shared drawing. Each user's alterations are captured and transmitted to the server computer. The server computer then broadcasts the alterations to each of the client computers associated with a collaborator. Each collaborator receives the alterations which are then incorporated into their copy of the shared drawing.

Each collaborator is provided with a set of drawing tools that can be used to add or delete lines and curves to the shared drawing. Each drawing tool has associated with it a specific style. The present technology captures the manner in which a user draws a stroke in order to emulate the style of the drawing tool used to draw the stroke.

A collaborator uses a pointing device to draw a stroke. Location data points are used to represent the path of the stroke. The location data points are then used to generate curve points representing the curve in accordance with a specified resolution. The curve points are used to display an image of the curve. In addition, curve descriptors are generated which are a compact representation of the shape of the curve. The curve descriptors are transmitted to the server computer for broadcast to the other collaborators. A collaborator receiving the curve descriptors translates them into curve points in order to display the image in the collaborator's virtual whiteboard.

The speed and pressure at which portions of the stroke are drawn is used to emulate the natural style of the drawing tool used in drawing the stroke. Time data points and/or pressure data points can be received from a pointing device used to generate the user's stroke. The time and/or data points can be used to emulate the drawing tool's natural style. The natural style of a drawing tool can be manifested in the varying color hues, texture, and thickness of the stroke.

BRIEF DESCRIPTION OF THE DRAWINGS

Additional objects and features of the invention will be more readily apparent from the following detailed description and appended claims when taken in conjunction with the drawings, in which:

FIG. 1 is a block diagram of a computer system incorporating the preferred embodiments of the present invention.

FIG. 2 shows a server computer system according to an embodiment of the present invention.

FIG. 3 is a schematic representation of an exemplary video display screen used in an embodiment of the present invention.

FIGS. 4A–4D are schematic representations of menu selection items used in an embodiment of the present invention.

FIG. 5 is a flow chart of the steps used to capture and transmit drawing alterations made during a collaborative design activity.

FIGS. 6A–6E are schematic representations of the steps used to trace the path of a curve drawn in a shared drawing.

FIG. 7 is a flow chart of the steps used to draw a curve in a shared drawing.

FIG. 8 illustrates the steps used by the image generation procedure in displaying an image.

FIGS. 9A–9C are schematic representations of steps used to erase a portion of a shared drawing.

FIG. 10 is a flow chart of the steps used to erase a portion of a shared drawing.

FIGS. 11A–11C are illustrations of curves used to generate a set of parallel lines.

FIG. 12 illustrates the steps used to generate a set of parallel lines.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

System Architecture

The technology of the present invention pertains to an apparatus and method for supporting a real-time computer-networked collaborative design activity. The collaborative design activity includes a community of users that communicate with one another through client computers interconnected via a public computer network such as the Internet. The communications between each participant includes all media forms that can be transmitted via a computer network. Examples of such media can include, but are not limited to, text, video, sound, and/or graphical images. A server computer is used to coordinate the transmission of the communications between each participant.

In an embodiment of the present invention, the collaborative design activity is a shared drawing activity between a community of users within the architecture engineering construction (AEC) industry. Each user utilizes a video display as a shared drawing space that can display one or more drawings. These drawings can include computer-aided design (CAD) drawings, construction blueprints, architectural drawings, computer-designed graphic images, and the like.

Each user or collaborator in the collaborative drawing session has its own copy of the shared drawing. Each user can edit the drawing using one or more graphic devices, such as but not limited to, a mouse or other pen pointing device. A collaborator can edit the shared drawing by adding a new drawing to it, change a part of the existing shared drawing, or perform other edit changes. The alterations to the drawing by each user are then captured and transmitted to the server computer. The server computer then broadcasts the alterations to each of the client computers associated with a collaborator. Each collaborator receives the alterations and incorporates them in their copy of the shared drawing.

In addition, each collaborator can send text messages to other collaborators. A dialogue or chat box is provided which allows real-time text-based messages to be communicated between one or more collaborators during the shared drawing session. In other embodiments of the present invention, messages including video and audio data can also be transmitted between the collaborators.

Referring to FIG. 1, a system **100** representing a preferred embodiment of the present invention includes a number of client computers **102** and one or more server computers **104** in communication via a communications link **106**. The communication link **106** generically refers to any type of wire or wireless link between computers, such as but not limited to a local area network, a wide area network, or a combination of networks. In a preferred embodiment of the present invention, the communications link **106** can be a network such as the Internet.

A client computer **102** includes a central processing unit (CPU) **108**, a user interface **110**, a memory **114**, and a communications interface **112**. The client computer **102** can be any type of computing device, such as but not limited to, desktop computers, workstations, laptops, and/or mainframe computers. One or more users (not shown) can be associated with each client computer **102**. The communications interface **112** is used to communicate with other client computers **102** as well as other system resources not relevant here.

The user interface **110** can consist of any type and number of input/output (I/O) devices, such as but not limited to, a video display device, printing devices, a keyboard, a mouse, pen pointing devices, graphic tablets (such as a WACOM pressure sensitive graphics tablet), and the like.

The memory **114** of the client computer **102** may be implemented as RAM (random access memory) or a combination of RAM and non-volatile memory such as magnetic disk storage. The memory **114** of the client computer **102** can contain the following:

- an operating system **116**;
- network access procedures **118**. In an embodiment of the present invention, the network access procedures **118** can be used to implement a communications protocol that is suitable for transmitting data through the Internet, such as the Transmission Control Protocol and the Internet Protocol (TCP/IP);
- an interactive drawing procedure **120** that manages the collaborative design activities between several client computers **102**;
- one or more session transcripts **136**. Each session transcript **136** stores a transcription of the events that occur during a collaborative design activity. A particular session transcript **136** can be replayed at a later point in time;
- a drawing repository **138** that stores one or more drawings **140**. A drawing **140** can be stored in any computer-generated image or graphic file format, such as but not limited to BMP, DXF, HPGL, DIB, TGA, GIF, TIF, PCX, JPG, and the like; and
- other procedures and data structures.

The interactive drawing procedure **120** can contain the following:

- one or more curve engine procedures **122**. Each curve engine procedure **122** is used to generate the curve descriptors and curve data points used to represent a curve;
- image generation procedure **124** that display images onto any type of I/O device, such as a video display device;
- session transcript procedures **126** that record portions of or the entire design activity session into a session transcript **136**. In addition, the session transcript procedures **126** enable a user to replay a particular session transcript **136** at a later point in time;
- conferencing procedures **128** that manage the exchange of communications between each collaborator during a design activity session;
- conversion procedures **130** that convert an incoming drawing stored in a particular file format into a format suitable for use by the interactive drawing procedure **120**;
- menu display procedures **132** that manage the drawing facilities provided to a user; and
- distance mapping procedures **134** that map a time and/or pressure value into an equivalent measure of distance;
- as well as other procedures and data structures.

Referring to FIG. 2, there is shown a server computer **104** including a central processing unit (CPU) **202**, a user interface **204**, a memory **208**, and a communications interface **206**. The memory **208** of the server computer **104** can contain the following:

- network access procedures **210**. In an embodiment of the present invention, the network access procedures **210** can be used to implement a communications protocol that is suitable for transmitting data through the Internet, such as the Transmission Control Protocol and the Internet Protocol (TCP/IP);
- an operating system **212**; and
- other procedures and data structures.

The operating system **212** of the server computer **104** utilizes broadcasting procedures **214** that manage the conferencing activities between the various client computers **102** engaged in the collaborative design activity. In an embodiment of the present technology, the UNIX operating system is used which supports a multi-user backbone (M-bone) or network ability that allows multiple client computers **102** to connect to a particular server computer **104**. However, it should be noted that the present invention is not limited to this particular type of operating system or broadcasting facility. Others can be used so long as they provide the same functionality.

The server's operating system procedures **212** utilizes a network port to support a number of communication streams that are used to transmit data between the client computers **102** engaged in the collaborative design activity. The server's operating system **212** receives transmissions from one or more client computers **102** and routes them to the intended client computers **102**. Initially, a client computer **102** sends a request to the server computer **104** to connect. The request contains the Internet address of each client computer **102** of a collaborator. The server computer **104** makes the connection and establishes a stream representing a connection between the client computer **102** with its network port. This stream allows communications to be transmitted rather rapidly between the server computer **104** and the particular client computer **102**.

Each collaborator participates in a collaborative drawing activity by utilizing the video display as a "virtual" whiteboard. The whiteboard can display any and all graphic images that are part of the collaborative drawing activity. FIG. **3** illustrates an exemplary screen display in accordance with an embodiment of the present invention. The interactive drawing procedure **120** displays the virtual whiteboard **302** along with a menu bar **304**, a dialogue box **306**, and a text input box **308**. The virtual whiteboard **302** can consist of one window illustrating a particular drawing **310**, can include multiple windows (not shown) with each window focusing on a particular aspect of the drawing **310**, or multiple windows (not shown) with each window displaying a different drawing. The dialogue box **306** is used to display text messages that are transmitted and received during a collaborative drawing session. The text input box **308** is used by a user to insert text messages that are distributed via the server computer **104** to the other collaborators of the collaborative drawing session.

Each collaborator can utilize a different type of video display device. For this reason, each collaborator's whiteboard is presumed to be of a standard size and is scaled, if necessary, to accommodate the dimensions of a particular collaborator's video display device.

Each collaborator can edit the drawing in a variety of ways using different tools **312** to draw different objects, lines, strokes, shapes, and the like. A menu display procedure **132** provides a number of facilities that enable the user to edit a drawing in an intended manner. FIGS. **4A–4D** illustrate an exemplary list of such facilities. For instance, the user can select a particular drawing tool **312** from a menu that includes, but is not limited to, the following tools: a fine pen, a wet brush, a watercolor brush, a pencil, a charcoal pen, an etch device, a straight edge, a crayon, a piece of chalk and/or a rollover. A user can use a particular tool to draw a stroke or text in its whiteboard. In addition, one or more templates of predefined shapes **314** or objects is provided for the user. The style of each stroke, shape, or text can be associated with one or more styles **318** and/or textures **316**. The textures **316** can represent various fill patterns. In an embodiment of the present invention, the textures **316** can represent materials used in the AEC industry. For instance, a diagonal cross-hatch texture can represent concrete, a orthogonal cross-hatch texture can represent ceramic title, and so forth.

The general architecture and processing associated with the interactive collaborative design activity has now been disclosed. Attention presently turns to the curve generating techniques associated with the present technology.

Curve Generating Method

In the collaborative drawing session of the present invention, each user can edit the shared drawing. Typically, a user edits the drawing by adding or deleting strokes to the shared drawing. A pointing device is used to trace the path of the user's stroke. The pointing device represents the path as a number of position data points. The position data points are then mapped into curve descriptors which are a concise representation of the Bezier curves that form the stroke. In addition, the stroke is also represented by a number of curve points that are used by the image generation procedure to display the stroke. The curve descriptors are transmitted to the server computer and broadcasted to the other collaborators.

In some instances, time and pressure data points are received along with the position data points. The time data points reflect the speed at which the stroke was generated and the pressure data points reflect the pressure used to generate the stroke. The time and pressure points are used to capture the natural style of the drawing tool used to draw the stroke. The natural style of a drawing tool is manifested by the varying color hues in the stroke, by the width of the stroke, and by the texture of stroke. For instance, segments of a stroke drawn fast are represented by a darker hue or texture than segments that are drawn slower. In addition, segments of a stroke that are drawn fast are often thinner than those drawn slowly. Likewise, the pressure of a stroke can affect the color hue, texture, and width characteristics. For example, segments of a stroke that are generated with heavy pressure can be reflected with a darker hue, darker texture and with a wider shape. By contrast, a lightly drawn stroke can be reflected with a lighter hue, lighter texture and with a thinner shape.

The characteristics of each drawing tool varies. As such, the time and pressure data points are used differently to reflect the natural style of a particular drawing tool. For example, pressure has an effect on the stroke that is generated by a water color brush but speed may not. The pressure used to draw a stroke with a water color brush will affect the width of the stroke but not the hue. By contrast, pressure and speed affect the style of a wet paint brush stroke. The more pressure that is used to draw the stroke, the thicker and darker the stroke becomes. Similarly, the faster the stroke is drawn, the thicker and darker the stroke becomes.

The curve generating method of the present technology traces the path of a user's stroke and generates a curve from it. The curve is represented as a set of curve points and as a set of curve descriptors. The curve points are used to display the image in accordance with user-defined directives. The user-defined directives can specify the curve's resolution and the curve characteristics relative to a particular drawing tool.

The curve descriptors define the shape of the curve. Three curve descriptors are used to represent a curve: a first descriptor represents a starting point of the curve; a second descriptor represents the shape of the curve; and a third descriptor represents an ending point of the curve. The curve descriptors are transmitted to the collaborators of a collaborative drawing session. The curve descriptors are transmitted rather than the position data points in order to minimize the amount of data that is transmitted between the collaborators. In this manner, the speed of the transmission is increased and relatively low bandwidth transmission links can be utilized without compromising the speed at which drawing updates are broadcast.

The curve generating method of the present technology differs from curve fitting techniques. Curve fitting techniques "fit" or generate a curve from a set of user-defined control points that describe the shape of the curve. The control points are not necessarily part of the curve. By contrast, the present technology receives a curve and translates it into another curve in accordance with one or more user-defined directives (i.e., resolution value and drawing tool).

Prior to discussing the details of the curve generating methods, a brief synopsis of the method is described first followed by a glossary of terms used in this document concerning the curve generating method. The curve generating method utilizes Bezier curve methods which are well-known in the art. A more detailed description of the Bezier curve methods can be found in Newman and Sproull, *Principles of Interactive Computer Graphics,* 2d edition, McGraw-Hill Book Company, which is hereby incorporated by reference as background information.

The curve generating method maps position data points received from a pointing device into a set of curve data points that define one or more Bezier curves. The curve engine **122** performs this mapping in real time and is sensitive to real-time processing constraints.

A stroke is represented as a series of second degree Bezier curves. Each Bezier curve is represented by a mathematical function q(u). The function q(u) is defined over a sat of evenly-distributed intervals represented by the vector u. Each interval u_j is a real number within the range between 0 to 1. The number of intervals, n, effects the resolution of the curve. The number of intervals can be either a user-defined value or determined from the curve engine **122**. In order to process the position data points in real time, the curve engine **122** may adjust the number of intervals in order to meet the demands of real-time processing.

Time and pressure data points can also be received from the pointing device. These data points are also transformed into a series of second degree Bezier curves in accordance with the same mathematical function q(u). The time and pressure Bezier curves are used by the image generation procedure **124** to generate an intended image of the curve.

To assist the reader, the following glossary of terms is used to describe the curve generating method:

Position data point: An (x, y) integer pair representing the logical location of the device pointer relative to the upper left hand corner of the video display device.

Time data point: An integer value within the range [0, 65535] that represents a time unit at which a corresponding position data point is generated. In an embodiment of the present invention, the time unit is expressed in milliseconds and represents the time differential from the immediately previous point.

Pressure data point: An integer value within the range [0,255] that represents the pressure associated with a corresponding position data point. In an embodiment of the present technology, 0 represents no pressure and 255 represents the heaviest amount of pressure.

Curve segment: A second degree Bezier curve, $q_1(u)$, that is represented by a vector, u, of location curve points, u_j, that are defined via the mathematical relation:

$$q_1(u_j) = \sum_{i=0}^{d} V_1(i) B_{d,i}(u_j) \tag{1}$$

where

d=2,

u_j represents a relative location along the curve. The values of u are real numbers u_n such that $0 \leqq u_n \leqq 1$, where n controls the resolution of the curve,

$B_{d,i} = u_j^i (1-u_j)^{d-i}$, and

$V_1(i)$ are location curve descriptors. $V_1(0)$ represents a starting point of the curve segment, $V_1(2)$ represents an ending point of the curve segment, $V_1(1)$ represents a shape descriptor represented in accordance with the following mathematical relation:

$$V_1(1) = \frac{q_1(u_k) - V_1(0)B_{2,0}(u_k) - V_1(2)B_{2,2}(u_k)}{B_{2,1}(u_k)} \tag{2}$$

where u_k is the interval that represents the furthest distance from the starting and ending points and can be determined via the following mathematical relation:

$$\max_{0<i<n}\{|x_0,y_0 - x_i,y_i| + |x_i,y_i - x_n,y_n|\} \tag{3}$$

where x_0,y_0 is the starting point of the curve segment, x_n,y_n is the ending point of the curve segment, and x_i,y_i are all other location curve points in the curve segment.

Stroke: The curve drawn by a user. A stroke is represented by a series of curve segments $q_{1,1}(u)$, $q_{1,2}(u)$, . . . , $q_{1,n}(u)$.

Location curve descriptors: Control points that define a curve segment and are defined above.

Time curve: A series of time segments representing a time dimension associated with the user drawn stroke.

Time segment: A second degree Bezier curve, $q_2(u)$, that is represented by a vector, u, of time curve points, u_j, that are defined via the mathematical relation:

$$q_2(u_j) = \sum_{i=0}^{d} V_2(i) B_{d,i}(u_j) \tag{4}$$

where

d=2,

u_j represents a relative location along the curve. The values of u are real numbers u_n such that $0 \leqq u_n \leqq 1$, where n controls the resolution of the curve,

$B_{d,i} = u_j^i (1-u_j)^{d-i}$, and

$V_2(i)$ are time curve descriptors. $V_2(0)$ represents a starting time point, $V_2(2)$ represents an ending time point, $V_2(1)$ represents a shape descriptor represented in accordance with the following mathematical relation:

$$V_2(1) = \frac{q_2(u_k) - V_2(0)B_{2,0}(u_k) - V_2(2)B_{2,2}(u_k)}{B_{2,1}(u_k)} \tag{5}$$

where u_k is the interval that represents the furthest distance from the starting and ending points and can be determined via the following mathematical relation:

$$\max_{0<i<n}\{|x_0,y_0 - x_i,y_i| + |x_i,y_i - x_n,y_n|\} \tag{6}$$

where x_0,y_0 is the starting time point, x_n,y_n is the ending time point, and x_i,y_i are all other time curve points in the curve segment.

Pressure curve: A series of pressure segments representing a pressure dimension associated with the user drawn stroke.

Pressure segment: A second degree Bezier curve, $q_3(u)$, that is represented by a vector, u, of pressure curve points, u_j, that are defined via the mathematical relation:

$$q_3(u_j) = \sum_{i=0}^{d} V_3(i) B_{d,i}(u_j) \tag{7}$$

where

d=2,

u_j represents a relative location along the curve. The values of u are real numbers u_n such that $0 \leqq u_n \leqq 1$, where n controls the resolution of the curve,

$B_{d,i}=u_j^i(1-u_j)^{d-i}$, and

$V_3(i)$ are pressure curve descriptors. $V_3(0)$ represents a starting pressure curve point, $V_3(2)$ represents an ending pressure curve point, $V_3(1)$ represents a shape descriptor represented in accordance with the following mathematical relation:

$$V_3(1)=\frac{q_3(u_k)-V_3(0)B_{2,0}(u_k)-V_3(2)B_{2,2}(u_k)}{B_{2,1}(u_k)} \tag{8}$$

where u_k is the interval that represents the furthest distance from the starting and ending points and can be determined via the following mathematical relation:

$$\max_{0<i<n}\{|x_0,y_0-x_i,y_i|+|x_i,y_i-x_n,y_n|\} \tag{9}$$

where x_0,y_0 is the starting pressure curve point, x_n,y_n is the ending pressure curve point, and x_i,y_i are all other pressure curve points in the curve segment.

FIG. 5 illustrates the steps used in the curve generating method of an embodiment of the present invention. During a collaborative drawing session, a collaborator can draw a stroke utilizing a pointing device. The pointing device represents the locations pointed to by the user as a set of position data points. The position data points are integer value pairs (x,y) that represent the logical location of the device pointer relative to the upper left hand corner of the video device. The curve generation procedure **122** receives the position data points in real-time and simultaneously generates curve points and curve descriptors (step **502**). The details of the generation of the curve points and curve descriptors will be discussed below.

The curve points are then used by the image generation procedure **124** to display the curve on the user's video display device or other I/O device (step **504**). The curve descriptors are formatted and transmitted to the server computer **104** for broadcast to other collaborators (step **506**). The server computer **104** receives this data and broadcasts it to the intended collaborators (step **508**). A collaborator receives the transmission, generates the appropriate curve points that correspond to the curve descriptors, and displays the new image (step **510**).

FIGS. 6A–6E illustrate the curve generating method. The user's stroke is represented by a series of position data points that are represented as x,y integer pairs. The position data points are partitioned into curve segments and a Bezier curve is defined for each curve segment. The Bezier curve consists of a number of curve points which are used to display the resulting image. The number of curve points is a function of the curve's resolution. Coarsely drawn curves are represented by a smaller number of curve points than smoothly drawn curves.

Referring to FIG. 6A, each integer pair (x,y) is received in real time by the curve engine **122**. The first point (x_0,y_0) is denoted as the starting curve point of the first curve segment. The starting point is also the first curve descriptor V(0). Each subsequent point is analyzed to determine whether it is part of the first curve segment. Referring to FIGS. 6B–6C, this is doe by determining the angle, θ, between the starting point, (x_0,y_0), and the previous data point (x_1,y_1) and the starting point, (x_0,y_0), and the current data point (x_2,y_2). If the angle θ exceeds a predetermined threshold, the previous data point (x_1,y_1) is made the ending point V(2) of one curve segment and the starting point V(0) of a new curve segment. The ending point V(2) is also a curve descriptor. In addition, a new curve segment is formed when the direction of the angle θ is opposite of the direction of the previously-calculated angle.

A third curve descriptor V(1) is generated to represent the shape or peak of the curve segment in accordance with equation (2) above. FIG. 6C illustrates the various curve segments formed from a user's stroke and their respective curve descriptors. The curve descriptors for each curve segment of a user's stroke is transmitted to the collaborators engaged in the drawing activity.

A Bezier curve is formulated for each defined curve segment. The Bezier curve is represented as a series of curve points. A curve segment is displayed as a series of lines that connect the curve points as shown in FIG. 6D. Thus, the number of curve points determines the resolution of the curve. The curve points are transmitted to the image generation procedure **124** which displays the resulting curve as illustrated in FIG. 6E.

In addition, time and pressure data points can be received from the pointing device along with the position data points. A time data point can be an integer pair representing a time value and an index. The time value can be within the range [0, 65535] and is expressed in milliseconds. The index represents the order of a time data point relative to other time data points. The time data points are mapped into a set of time curve points and time curve descriptors. The time curve points are used along with the location data points to display the image in accordance with the natural style of a particular drawing tool. The time curve descriptors are transmitted to other collaborators along with the location curve descriptors and/or pressure curve descriptors in order to display the curve in the collaborator's whiteboard.

Similarly, a pressure data point can be an integer pair representing a pressure value and an index. The pressure data point value can be within the range [0, 255]. The index represents the order of a pressure point relative to other pressure points. The pressure data point is mapped into a set of pressure curve points and pressure curve descriptors. The pressure curve points are used along with the location curve points and/or time curve points to display the image in accordance with the natural style of a particular drawing tool. The pressure curve descriptors are transmitted to other collaborators along with the location and/or time curve descriptors in order to display the curve in the collaborator's whiteboard.

FIG. 7 illustrates the steps used in generating a curve in an embodiment of the present technology. It should be noted that a user's stroke can include lines as well as curves and that the curve generation engine **122** can accommodate lines as well as curves. The following discussion focuses on curves and can be easily modified by one skilled in the art to accommodate lines.

Further, the following discussion pertains to an embodiment where the pointing device retrieves location, time, and pressure data points from a user's stroke. In alternate embodiments, the pointing device would retrieve any combination of location data points and either time or pressure data points. One skilled in the art would be able to easily modify the discussed methodology to accommodate these alternate embodiments. In yet another embodiment, the time data points need not be extracted from the pointing device. The time data points can be calculated by the curve generation engine **122** using the location data points. This can be performed by calculating the distance between two successive location data points and mapping the distance into an appropriate time unit. In addition, other methodologies can be used to calculate time data points from the location data points.

Referring to FIG. 7, the curve engine **122** receives in real time a number of data points (step **520**) represented as follows:

I_i: location data point represented as (x,y).

t_i: time data point represented as (z_1, z_2) where z_1 is an integer value within the range [0,65535] and a time unit expressed in milliseconds and where z_2 represents the index of the time data point relative to other time data points.

p_i: pressure data point represented as (p_1, p_2) where p_1 is an integer value within the range [0, 255] and where p_2 represents the index of the pressure date point relative to other pressure data points.

$V_{1j}(0)$, $V_{1j}(1)$, $V_{1,j}(2)$: curve descriptors for curve segment j.

$V_{2j}(0)$, $V_{2j}(1)$, $V_{2j}(2)$: curve descriptors for time curve segment j.

$V_{3j}(0)$, $V_{3j}(1)$, $V_{3j}(2)$: curve descriptors for pressure curve segment j.

The curve engine procedure **122** determines if the received data points are the first set of data points (step **522**). In this case (step **522**-Y), the start and end curve descriptors V(0) and V(2) for each data point type is initialized as follows (step **524**):

$$V_{1j}(0)=V_{1j}(2)=I_i;\ V_{2j}(0)=V_{2j}(2)=t_i;\ V_{3j}(0)=V_{3j}(2)=p_i. \tag{10}$$

If the received data points are the second set of data points (step **526**), the end curve descriptors are set to the respective second data point as follows (step **528**):

$$V_{1j}(2)=I_i;\ V_{2j}(2)=t_i;\ V_{3j}(2)=p_i. \tag{11}$$

For each subsequent set of data points, the direction of the curve and its growth is determined relative to the starting point or first curve descriptor (step **530**). This is performed by determining the angle between two secant lines representing the previous location data point and the current location data point (see FIG. 6B). This can be determined mathematically as follows:

$$\theta=\arctan\left((|y_1-y_0|-|y_2-y_0|)-(|x_1-x_0|-|x_2-x_0|)\right) \tag{12}$$

In addition the direction of the angle θ is determined and compared with the direction of the immediately preceding calculated angle. The direction of the angle θ is expressed as being either a clockwise direction or as a counter clockwise direction.

The computed angle θ and its direction is used to determine if the received location data point is past a critical point defining the current curve segment (step **532**). This is determined by testing the computed angle θ against a user-defined threshold. In an embodiment of the present technology this threshold is between the range (π/4, π/10). If the direction of the angle is opposite to the previously calculated angle, this will indicate that the received location data point is outside the realm of the current curve segment.

If it is determined that the received data points define a new curve segment (step **532**-Y), the curve descriptors V(0) and V(2) are initialized to the respective data points as follows (step **534**):

$$V_{1j+1}(2)=I_i;\ V_{2j+1}(2)=t_i;\ V_{3,j+1}(2)=p_i.$$

$$V_{1j+1}(0)=I_{i-1};\ V_{2i+1}(0)=t_{i-1};\ V_{3,j+1}(0)=p_{i-1}. \tag{13}$$

Otherwise (step **532**-N), the appropriate curve descriptors and curve points are calculated (step **536**). The $V_1(1)$, $V_2(1)$, $V_3(1)$ curve descriptors are calculated as described in equations (2), (5), and (8) and the curve points $q_1(u)$, $q_2(u)$, and $q_3(u)$ are calculated as well as described in equations (1), (4), and (7).

Steps **520**–**536** are repeated until no more data points are received by the curve engine procedure **122** from the pointing device. This typically signifies the fact that the user has completed the stroke by removing the pointing device from the visual display device.

Referring to FIG. **5**, upon the completion of the above mentioned curve generating method, the curve descriptors are transmitted to the server computer for broadcast to the other collaborators (step **506**). The transmission can be formatted to include the following: drawing tool indicator, location curve descriptors, time curve descriptors, and pressure curve descriptors. The drawing tool indicator specifies the particular drawing tool used to make the stroke.

When a collaborator receives the curve descriptors, the curve engine **122** transforms the curve descriptors into a respective set of curve points in accordance with equations (1), (4), and (7). The curve points are then transmitted to the image generation procedure **124** which displays the curve in the collaborator's whiteboard.

Image Generating Method

Referring to FIG. **8**, the image generation procedure **124** can receive three sets of curve points: a set of location curve points $q_{1,1}(u) \ldots q_{1,n}(u)$; a set of time curve points $q_{2,1}(u) \ldots q_{2,n}(u)$; and a set of pressure curve points $q_{3,1}(u) \ldots q_{3,n}(u)$ which it uses to display the associated image **556**. The image generation procedure **124** can utilize a scale procedure **550** to map the location curve points $q_{1,1}(u) \ldots q_{1,n}(u)$ into suitable coordinates for the particular collaborator's visual display device. In addition, each drawing tool has associated with it one or more scale tables **552**, **554** that are used to map the time curve points $q_{2,1}(u) \ldots q_{2,n}(u)$ and pressure curve points $q_{3,1}(u) \ldots q_{3,n}(u)$ into suitable color and dimension attributes associated with the video display device.

The general curve and image generating methods associated with the collaborative design activity have now been disclosed. The present technology provides two additional capabilities that will now be discussed. Specifically the capability for erasing a displayed curve and the capability of automatically generating parallel lines from a single user's stroke.

Erasure

During a collaborative drawing session, a user can erase any lines or curves in the shared drawing. For illustration purposes, the situation where a user specifies two points indicating where a curve is to be erased will be described. However, it should be noted that the present technology can erase a line or curve in any number of locations.

Referring to FIGS. **9**–**10**, a user indicates with the pointing device two points **560**, **562** on the curve that define an area of the curve that is to be erased. These points **560**, **562** are received by the curve generating procedure **122** (step **570**). The curve generating procedure **122** determines the effected curve segments and their associated curve descriptors (step **572**). For the curve segment corresponding to the first erasure data point (step **574**-Y), the curve segment's end point is adjusted to be first erasure data point. Likewise, for the curve segment corresponding to the second erasure data point (step 574-N), the curve segment's starting point is adjusted to be the second erasure data point (step 578). The appropriate curve data points $q_1(u)$, $q_2(u)$, $q_3(u)$ representing the affected curve segments are recalculated as well as the V(1) curve descriptors in accordance with the aforementioned equations (step 580) (see FIG. 9B). Also, any curve segments located entirely between the deletion points 560, 562 are deleted (step 580).

The curve data points are then transmitted to the image generation procedure 124 to display the new image (see FIG. 9C). The curve descriptors for both of the newly generated curves are transmitted to the server computer 104 for broadcast to each collaborator.

Parallel Curves

Another facility of the present technology is the capability to automatically generate parallel lines or curves from a user's stroke. For illustration purposes, the following discussion will focus on the generation of two parallel curves. However, one skilled in the art can easily modify the described technique to generate parallel lines and multiple parallel lines and curves.

A user traces the path of a curve with a pointing device. The curve is represented as a series of position data points. Each of the two parallel curves will be generated at an equal distance above and below the user's stroke. Time and/or pressure data points associated with the drawn curve are used to determine the distance between the parallel curves (see FIG. 11A). In an alternate embodiment, the time data points need not be received from the pointing device. The time data points can be inferentially calculated by measuring the distance between two position data points as previously described above. In addition, the time and/or pressure data points can also be used to affect the width, texture, and color hue of each of the parallel curves as previously described above.

Referring to FIG. 12, a set of position data points (x_i,y_i) and time and/or pressure data points are received from an I/O device 582. The time and/or pressure data points are mapped via a distance mapping procedure 134 into an appropriate distance measure d_i. The distance is divided in half. The distance $(d_i/2)$ and the position data points are transmitted simultaneously to two curve engines 122. The first curve engine 122*a* generates the curve points and curve descriptors for the top parallel curve and the second curve engine 122*b* generates the curve points and curve descriptors for the bottom parallel curve. The curve points for both curves are then transmitted to the image generation procedure 124 which displays the parallel curves 584, 586 as shown in FIG. 12B. The curve descriptors are transmitted to the server computer 104 for broadcast to the other collaborators.

In an alternate embodiment, a user's stroke can be mapped into an image including two parallel curves where the area between the parallel curves includes a user-defined texture as shown in FIG. 11C. The above mentioned steps for generating parallel curves is used to generate two parallel curves. Next, a user-defined texture is drawn in the area between the curves. This texture can also be affected by the time and pressure data points associated with the user drawn stroke. Curve descriptors are computed for this set of parallel curves and are transmitted along with the texture data to the server computer 104 for broadcast to the other collaborators.

Alternate Embodiments

While the present invention has been described with reference to a few specific embodiments, the description is illustrative of the invention and is not to be construed as limiting the invention. Various modifications may occur to those skilled in the art without departing from the true spirit and scope of the invention as defined by the appended claims.

The present invention is not limited to the computer system described in reference to FIG. 1. It may be practiced without the specific details and may be implemented in various configurations, or makes or models of distributed computing systems, tightly-coupled processors or in various configurations of loosely-coupled microprocessor systems.

Further, the method and system described hereinabove is amenable for execution on various types of executable mediums other than a memory device such as a random access memory. Other types of executable mediums can be used, such as but not limited to, a computer readable storage medium which can be any memory device, compact disc, or floppy disk.

The present technology has been described with reference to curves and lines. However, the present technology is amenable to any two-dimensional graphic image. Furthermore, one skilled in the art can modify the above discussed technology to accommodate any n-dimensional graphic image.

What is claimed is:

1. A method for generating a drawing in a distributed computing system including at least two client computers and at least one server computer, the method comprising the steps of:

(a) providing one or more drawing tools for use by users associated with the client computers to draw a curve, each drawing tool having one or more style attributes associated therewith;

(b) at a first one of the client computers, receiving a plurality of location data points representing a path of a curve drawn with one of the drawing tools;

(c) at the first client computer, mapping the location data points into a plurality of location curve data points, the location curve data points used to display the curve in accordance with a specified image resolution;

(d) utilizing the location curve data points to display the curve in accordance with the drawing tool used to draw the curve;

(e) at the first client computer, generating one or more sets of location curve descriptors, the location curve descriptors representing a compact representation of the location curve data points; and

(f) transmitting the location curve descriptors from the first client computer to the server computer for retransmission to at least a second one of the client computers;

(g) at the second client computer, converting the location curve descriptors into a plurality of location curve data points, the location curve data points used to display the curve in accordance with a specified resolution and in accordance with a specified drawing tool;

wherein each step performed at the first client computer is also performed at the second client computer and each step performed at the second client computer is also performed at the first client computer so as to enable the users at the first and second client computers to cooperatively generate the drawing.

2. A method for generating a curve in a distributed computing system including at least one client computer and at least one server computer, the method comprising the steps of:

(a) providing one or more drawing tools for use by users associated with the client computers to draw a curve, each drawing tool having one or more style attributes associated therewith;

(b) receiving a plurality of location data points representing a path of a curve drawn with one of the drawing tools;

(c) mapping the location data points into a plurality of location curve data points, the location curve data points used to display the curve in accordance with a specified image resolution; and

(d) utilizing the location curve data points to display the curve in accordance with the style attributes of the drawing tool used to draw the curve;

step (b) further including:

receiving a plurality of temporal data points representing a speed at which portions of the curve was drawn;

step (c) further including:

mapping the temporal data points into a plurality of temporal curve points, each of the temporal curve points associated with a corresponding curve data point; and

step (d) further including:

utilizing the temporal data points to display the curve in accordance with the drawing tool used to draw the curve.

3. The method of claim 2, wherein the temporal data points are calculated from a distance between consecutive location data points.

4. A method for generating a curve in a distributed computing system including at least one client computer and at least one server computer, the method comprising the steps of:

(a) providing one or more drawing tools for use by users associated with the client computers to draw a curve, each drawing tool having one or more style attributes associated therewith;

(b) receiving a plurality of location data points representing a path of a curve drawn with one of the drawing tools;

(c) mapping the location data points into a plurality of location curve data points, the location curve data points used to display the curve in accordance with a specified image resolution; and

(d) utilizing the location curve data points to display the curve in accordance with the drawing tool used to draw the curve;

step (b) further including:

receiving a plurality of pressure data points representing a pressure at which portions of the curve was drawn;

step (c) further including:

mapping the pressure data points into a plurality of pressure curve points, each pressure curve point associated with a corresponding curve data point; and

step (d) further including:

utilizing the pressure data points to display the curve in accordance with the drawing tool used to draw the curve.

5. The method of claim 1,

the style attributes including width attributes and color attributes.

6. The method of claim 2,

generating one or more sets of temporal curve descriptors, the temporal curve descriptors representing a compact representation of the temporal curve data points; and

transmitting the temporal curve descriptors to the server computer for broadcast to one or more of the client computers.

7. The method of claim 6,

for each of the receiving client computers, converting the temporal curve descriptors into a plurality of temporal curve data points, the temporal curve data points used to display the curve in accordance with a specified resolution and in accordance with a specified drawing tool.

8. The method of claim 4,

generating one or more sets of pressure curve descriptors, the pressure curve descriptors representing a compact representation of the pressure curve data points; and

transmitting the pressure curve descriptors to the server computer for broadcast to one or more of the client computers.

9. The method of claim 8,

for each of the receiving client computers, converting the pressure curve descriptors into a plurality of pressure curve data points, the pressure curve data points used to display the curve in accordance with a specified resolution and in accordance with a specified drawing tool.

10. An interactive drawing apparatus, comprising:

one or more client computers, each said client computer including:

a memory for storing a graphical drawing;

an input mechanism for receiving one or more hand-drawn shapes applied to said graphical drawing;

at least one curve engine for partitioning each said hand-drawn shape into a plurality of adjacent curves, each said adjacent curve having at least one boundary, each said boundary determined based on changes in direction of said hand-drawn shape meeting predefined threshold criteria, said curve engine representing each said adjacent curve as a set of control points defining said curve; and

at least one server computer, coupled by one or more communication links to the client computers, the at least one server computer including a communication mechanism for receiving said control points and transmitting said control points to one or more of said client computers.

11. The apparatus of claim 10,

said curve engine generating a set of location curve points associated with said control points, said set of location curve points used to display said hand-drawn shape in accordance with a predetermined criteria;

each said client computer including:

an output device for displaying said graphical drawing; and

an image generation procedure that receives said location curve points and displays said associated hand-drawn shape in accordance with said location curve points as part of said graphical drawing on said output device.

12. The apparatus of claim 10,

each said client computer including:

a conferencing procedure for managing an exchange of communications between one or more of said client computers.

13. The apparatus of claim 10,

each said client computer including:

a conversion procedure that imports a received drawing associated with a first format into said drawing associated with a second format.

14. The apparatus of claim 10,

each said client computer including:

a session transcript procedure that records alterations made to said drawing by each of said client computers.

15. A computer program product for use in conjunction with a computer system, the computer program product comprising a computer readable storage medium and a computer program mechanism embedded therein, the computer program mechanism comprising:

a graphical drawing;

an input mechanism for receiving one or more hand-drawn shapes applied to said graphical drawing;

at least one curve engine for partitioning each said hand-drawn shape into a plurality of adjacent curves, each said adjacent curve having at least one boundary, each said boundary determined based on changes in direction of said hand-drawn shape meeting predefined threshold criteria, said curve engine representing each said adjacent curve as a set of control points defining said curve; and

a broadcast mechanism for transmitting said control points to one or more of said client computers.

16. The computer program product of claim 15,

said curve engine generating a set of location curve points associated with said control points, said set of location curve points used to display said hand-drawn shape in accordance with a predetermined criteria;

an output device for displaying said graphical drawing; and

an image generation procedure that applies said location curve points to said graphical drawing on said output device.

17. The computer program product of claim 15,

a conferencing procedure for managing an exchange of communications between one or more of said client computers.

18. The computer program product of claim 15,

a conversion procedure that converts a received drawing associated with a first format into a second format associated with said graphical drawing.

19. The computer program product of claim 15,

a session transcript procedure that records alterations made to said graphical drawing by each of said client computers.

20. A computer-implemented method for generating and displaying a curved image, said method comprising the steps of:

(a) receiving a hand-drawn shape of a curved image;

(b) obtaining a distance measurement d from said hand-drawn shape;

(c) providing a set of parallel curved images, each of said parallel curved images represented as a plurality of adjacent curves, each said adjacent curve having at least one boundary, each said boundary determined based on changes in direction of said hand-drawn shape meeting a predefined threshold, a first one of said parallel curved images located at a distance d/2 above said shape and a second one of said parallel curved images located at a distance d/2 below said shape; and

(d) displaying each of said parallel curved images.

21. The method of claim 20,

wherein said hand-drawn shape is received as a plurality of position data points;

said step (b) further comprising the steps of:

(i) obtaining a plurality of temporal data points, each said temporal data point associated with a respective one of said position data points; and

(ii) mapping each of said temporal data points into a corresponding distance data point, said distance data point including a specified location for positioning an associated position data point;

said step (c) further comprising the step of:

(i) utilizing said position data points and said distance data points to generate said set of parallel curved images.

22. The method of claim 21,

said step (b)(i) further comprising the step of:

computing a temporal data point for each of said position data points, each said temporal data point representing a distance between a current position data point and an immediately preceding position data point.

23. The method of claim 20,

wherein said hand-drawn shape is received as a plurality of position data points;

said step (b) further comprising the steps of:

(i) receiving a plurality of pressure data points, each said pressure data point associated with a respective one of said position data points; and

(ii) mapping each of said pressure data points into a corresponding distance data point, said distance data point including a specified location for positioning an associated position data point; and

said step (c) further comprising the step of:

(i) utilizing said position data points and said distance data points to generate said set of parallel curved images.

24. An apparatus for creating a drawing, comprising:

one or more drawing tools for use by users of a client computer to draw a curve, each drawing tool having one or more style attributes associated therewith;

an input device for receiving a plurality of location data points representing a path of a curve drawn with one of the drawing tools;

a curve engine, coupled to the input device, for mapping the location data points into a plurality of location curve data points, the location curve data points used to display the curve in accordance with a specified image resolution;

a display device for displaying said freehand sketched input in accordance with the style attributes;

an image generation mechanism, coupled to the curve engine and display device, for mapping the location data points into an image of a curve in accordance with the drawing tool used to draw the curve;

the curve engine generating one or more sets of location curve descriptors, the location curve descriptors representing a compact representation of the location curve data points; and

a communication device for transmitting the location curve descriptors from the client computer to a server computer for retransmission to at least a second client computer, and for receiving location curve descriptors from the second client computer via the server computer;

the image generation mechanism including a mechanism for converting the location curve descriptors received via communication device into a plurality of location curve data points, and using the location curve data points to display the curve in accordance with a specified resolution and in accordance with a specified one of the drawing tools.

25. An apparatus for creating a drawing, comprising:

one or more drawing tools for use by users of a client computer to draw a curve, each drawing tool having one or more style attributes associated therewith;

an input device for receiving a freehand sketched input, the freehand sketched input including a plurality of location data points representing a path of a curve drawn with one of the drawing tools;

a curve engine, coupled to the input device, for mapping the location data points into a plurality of location curve data points, the location curve data points used to display the curve in accordance with a specified image resolution;

a display device for displaying said freehand sketched input in accordance with the style attributes;

an image generation mechanism, coupled to the curve engine and display device, for mapping the location data points into an image of a curve in accordance with the drawing tool used to draw the curve;

the curve engine generating one or more sets of location curve descriptors, the location curve descriptors representing a compact representation of the location curve data points;

wherein

the received freehand sketched input includes a plurality of temporal data points representing a speed at which portions of the curve was drawn;

the curve engine maps the temporal data points into a plurality of temporal curve points, each of the temporal curve points associated with a corresponding first location curve data point; and

the image generating mechanism utilizes the temporal data points to display the curve in accordance with the drawing tool used to draw the curve.

26. The apparatus of claim 25, wherein the temporal data points are calculated from a distance between consecutive location data points.

27. An apparatus for creating a drawing, comprising:

one or more drawing tools for use by users of a client computer to draw a curve, each drawing tool having one or more style attributes associated therewith;

an input device for receiving a freehand sketched input, the freehand sketched input including a plurality of location data points representing a path of a curve drawn with one of the drawing tools;

a curve engine, coupled to the input device, for mapping the location data points into a plurality of location curve data points, the location curve data points used to display the curve in accordance with a specified image resolution;

a display device for displaying said freehand sketched input in accordance with the style attributes;

an image generation mechanism, coupled to the curve engine and display device, for mapping the location data points into an image of a curve in accordance with the drawing tool used to draw the curve;

the curve engine generating one or more sets of location curve descriptors, the location curve descriptors representing a compact representation of the location curve data points;

wherein

the received freehand sketched input includes a plurality of pressure data points representing a pressure at which portions of the curve was drawn;

the curve engine maps the pressure data points into a plurality of pressure curve points, each pressure curve point associated with a corresponding curve data point; and

the image generating mechanism utilizes the pressure data points to display the curve in accordance with the drawing tool used to draw the curve.

* * * * *